(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2002－159004
（P2002－159004A）
(43)公開日　平成14年５月31日(2002.5.31)

(51)Int.Cl.[7]	識別記号	ＦＩ		テーマコード（参考）
Ｈ０４Ｎ　7/24		Ｃ１１Ｂ　20/10	３１１	５Ｃ０５３
Ｇ１１Ｂ　20/10	３１１	Ｈ０３Ｍ　7/30	Ｚ	５Ｃ０５９
Ｈ０３Ｍ　7/30		Ｈ０４Ｎ　7/13	Ｚ	５Ｄ０４４
Ｈ０４Ｎ　5/92		5/92	Ｈ	５Ｊ０６４

審査請求　未請求　請求項の数12　ＯＬ　（全 72 頁）

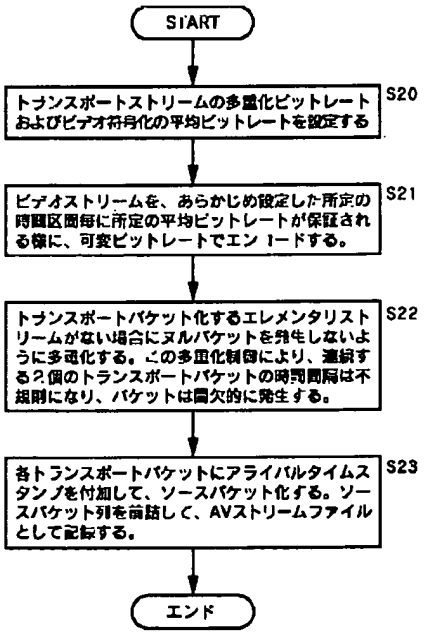
(21)出願番号　特願2001－112756(P2001－112756)
(22)出願日　平成13年４月11日(2001.4.11)
(31)優先権主張番号　特願2000－183770(P2000－183770)
(32)優先日　平成12年４月21日(2000.4.21)
(33)優先権主張国　日本（ＪＰ）
(31)優先権主張番号　特願2000－268042(P2000－268042)
(32)優先日　平成12年９月５日(2000.9.5)
(33)優先権主張国　日本（ＪＰ）

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番35号
(72)発明者　加藤　元樹
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(72)発明者　浜田　俊也
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(74)代理人　100082131
弁理士　稲本　義雄

最終頁に続く

(54)【発明の名称】　符号化装置および方法、記録媒体、並びにプログラム

(57)【要約】
【課題】　ディスク内に記録されている情報を確認、所望の情報の検索を、簡便に行えるようにする。
【解決手段】　ステップＳ２０で、トランスポートストリームの多重化ビットレートTS_recording_rateおよびビデオ符号化の平均ビットレートが設定される。ステップＳ２１で、ビデオストリームを、あらかじめ設定した所定の時間区間毎に所定の平均ビットレートが保証される様に、可変ビットレートでエンコードするようにビデオエンコーダが制御される。ステップＳ２２で、トランスポートパケット化するエレメンタリストリームがない場合にヌルパケットを発生しないようにマルチプレクサが制御される。ステップＳ２３で、各トランスポートパケットにアライバルタイムスタンプを付加して、ソースパケット化するように、ソースパケッタイザ１９が制御される。



AVストリームが時間経過とAVストリームのデータバイト量との関係が、比例することを保証する符号化モード（time_controlled_flag=1）において、ビデオを可変ビットレート符号化して、AVストリームを記録する動作を説明するフローチャート

【特許請求の範囲】
【請求項１】　映像データを符号化する符号化装置において、
前記映像データを可変レートにより符号化する符号化器と、
時間経過に対して映像符号化データ量がほぼ比例するように制御する制御部とを有することを特徴とする符号化装置。
【請求項２】　前記制御部は、単位時間あたりの映像符号化データ発生量が所定量に満たないときにはスタッフィングバイトを符号化するよう制御することを特徴とする請求項１に記載の符号化装置。
【請求項３】　前記制御部は、各々のピクチャの符号化の際に発生するデータ量に応じてスタッフィングバイトを符号化するか否かを判定することを特徴とする請求項２に記載の符号化装置。
【請求項４】　前記制御部は、ＶＢＶバッファがオーバーフローしないようにスタッフィングバイトを符号化するよう制御することを特徴とする請求項２に記載の符号化装置。
【請求項５】　前記制御部は、時間経過に対して符号化データ量がほぼ比例するように符号化する符号化モードと通常の符号化モードのどちらか一方で符号化するように制御することを特徴とする請求項１に記載の符号化装置。
【請求項６】　前記制御部は、前記符号化モードが、時間経過に対して符号化データ量がほぼ比例するように符号化するモードか否かを示す付加情報を生成することを特徴とする請求項５に記載の符号化装置。
【請求項７】　映像データを符号化する符号化装置の符号化方法において、
前記映像データを可変レートにより符号化する符号化ステップと、
時間経過に対して映像符号化データ量がほぼ比例するように制御する制御ステップとを含むことを特徴とする符号化方法。
【請求項８】　映像データを符号化する符号化装置を制御するプログラムにおいて、
前記映像データを可変レートにより符号化する符号化ステップと、
時間経過に対して映像符号化データ量がほぼ比例するように制御する制御ステップとを含むことを特徴とするコンピュータが読み取り可能なプログラムが記録されている記録媒体。
【請求項９】　映像データを符号化する符号化装置を制御するコンピュータに、
前記映像データを可変レートにより符号化する符号化ステップと、
時間経過に対して映像符号化データ量がほぼ比例するように制御する制御ステップとを実行させるプログラム。
【請求項１０】　映像データが記録されている記録媒体において、
前記映像データと、前記映像データに対応するオーディオデータを含むAVストリームファイルと、
前記AVストリームファイルの記録モードを示すフラグとが記録されていることを特徴とする記録媒体。
【請求項１１】　前記フラグは、time_controlled_flagであることを特徴とする請求項１０に記載の記録媒体。
【請求項１２】　前記フラグは、記録してからの時間経過に対してファイルサイズが比例するようにして記録されるモードであることを示すことを特徴とする請求項１１に記載の記録媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は符号化装置および方法、記録媒体、並びにプログラムに関し、特に、記録媒体に記録されているデータの内容の管理情報をファイル化して記録する符号化装置および方法、記録媒体、並びにプログラムに関する。
【０００２】
【従来の技術】近年、記録再生装置から取り外し可能なディスク型の記録媒体として、各種の光ディスクが提案されつつある。このような記録可能な光ディスクは、数ギガバイトの大容量メディアとして提案されており、ビデオ信号等のAV(Audio Visual)信号を記録するメディアとしての期待が高い。この記録可能な光デイスクに記録するデジタルのAV信号のソース（供給源）としては、CSデジタル衛星放送やBSデジタル放送があり、また、将来はデジタル方式の地上波テレビジョン放送等も提案されている。
【０００３】ここで、これらのソースから供給されるデジタルビデオ信号は、通常MPEG（Moving Picture Experts Group）２方式で画像圧縮されているのが一般的である。また、記録装置には、その装置固有の記録レートが定められている。従来の民生用映像蓄積メディアで、デジタル放送由来のデジタルビデオ信号を記録する場合、アナログ記録方式であれば、デジタルビデオ信号をデコード後、帯域制限をして記録する。あるいは、MPEG１ Video、MPEG２ Video、DV方式をはじめとするデジタル記録方式であれば、１度デコードされた後に、その装置固有の記録レート・符号化方式で再エンコードされて記録される。
【０００４】しかしながら、このような記録方法は、供給されたビットストリームを１度デコードし、その後で帯域制限や再エンコードを行って記録するため、画質の劣化を伴う。画像圧縮されたデジタル信号の記録をする場合、入力されたデジタル信号の伝送レートが記録再生装置の記録レートを超えない場合には、供給されたビットストリームをデコードや再エンコードすることなく、そのまま記録する方法が最も画質の劣化が少ない。ただ

し、画像圧縮されたデジタル信号の伝送レートが記録媒体としてのディスクの記録レートを超える場合には、記録再生装置でデコード後、伝送レートがディスクの記録レートの上限以下になるように、再エンコードをして記録する必要はある。

【０００５】また、入力デジタル信号のビットレートが時間により増減する可変レート方式によって伝送されている場合には、回転ヘッドが固定回転数であるために記録レートが固定レートになるテープ記録方式に比べ、１度バッファにデータを蓄積し、バースト的に記録ができるディスク記録装置が記録媒体の容量をより無駄なく利用できる。

【０００６】以上のように、デジタル放送が主流となる将来においては、データストリーマのように放送信号をデジタル信号のまま、デコードや再エンコードすることなく記録し、記録媒体としてディスクを使用した記録再生装置が求められると予測される。

【０００７】

【発明が解決しようとする課題】上述したように、記録媒体の容量が増大することにより、その記録媒体には、多くのデータ（この場合、番組に関する映像や音声など）が記録できるようになる。従って、１枚のディスクに多くの番組が記録されることになり、ユーザが、それらのディスク内に記録されている多くの番組から視聴したい１番組を選択するといったような操作が煩雑になってしまう。そこで、ユーザがディスクの再生時に、簡便に記録されているデータを確認し、所望の番組（データ）が選択できるようにする必要があるといった課題があった。

【０００８】本発明はこのような状況に鑑みてなされたものであり、記録媒体に記録されているデータの内容の管理情報をファイル化して記録する事により、記録媒体に記録されているデータ内容、および、再生情報を適切に管理することができるようにすることを目的とする。

【０００９】

【課題を解決するための手段】本発明の符号化装置は、映像データを可変レートにより符号化する符号化器と、時間経過に対して映像符号化データ量がほぼ比例するように制御する制御部とを有することを特徴とする。

【００１０】前記制御部は、単位時間あたりの映像符号化データ発生量が所定量に満たないときにはスタッフィングバイトを符号化するよう制御することができる。

【００１１】前記制御部は、各々のピクチャの符号化の際に発生するデータ量に応じてスタッフィングバイトを符号化するか否かを判定することができる。

【００１２】前記制御部は、ＶＢＶバッファがオーバーフローしないようにスタッフィングバイトを符号化するよう制御することができる。

【００１３】前記制御部は、時間経過に対して符号化データ量がほぼ比例するように符号化する符号化モードと通常の符号化モードのどちらか一方で符号化するように制御することができる。

【００１４】前記制御部は、符号化モードが、時間経過に対して符号化データ量がほぼ比例するように符号化するモードか否かを示す付加情報を生成することができる。

【００１５】本発明の符号化方法は、映像データを可変レートにより符号化する符号化ステップと、時間経過に対して映像符号化データ量がほぼ比例するように制御する制御ステップとを含むことを特徴とする。

【００１６】本発明の記録媒体のプログラムは、映像データを可変レートにより符号化する符号化ステップと、時間経過に対して映像符号化データ量がほぼ比例するように制御する制御ステップとを含むことを特徴とする。

【００１７】本発明のプログラムは、映像データを可変レートにより符号化する符号化ステップと、時間経過に対して映像符号化データ量がほぼ比例するように制御する制御ステップとを実行させる。

【００１８】本発明の符号化装置および方法、記録媒体、並びにプログラムにおいては、映像データが可変レートによって符号化され、時間経過に対して映像符号化データ量がほぼ比例するように制御される。

【００１９】本発明の記録媒体は、映像データと、映像データに対応するオーディオデータを含むAVストリームファイルと、AVストリームファイルの記録モードを示すフラグとが記録されていることを特徴とする。

【００２０】前記フラグは、time_controlled_flagとすることができる。

【００２１】前記フラグは、記録してからの時間経過に対してファイルサイズが比例するようにして記録されるモードであることを示すようにすることができる。

【００２２】本発明の記録媒体においては、映像データと、映像データに対応するオーディオデータを含むAVストリームファイルと、AVストリームファイルの記録モードを示すフラグとが記録されている。

【００２３】

【発明の実施の形態】以下に、本発明の実施の形態について、図面を参照して説明する。図１は、本発明を適用した記録再生装置１の内部構成例を示す図である。まず、外部から入力された信号を記録媒体に記録する動作を行う部分の構成について説明する。記録再生装置１は、アナログデータ、または、デジタルデータを入力し、記録することができる構成とされている。

【００２４】端子１１には、アナログのビデオ信号が、端子１２には、アナログのオーディオ信号が、それぞれ入力される。端子１１に入力されたビデオ信号は、解析部１４とAVエンコーダ１５に、それぞれ出力される。端子１２に入力されたオーディオ信号は、AVエンコーダ１５に出力される。解析部１４は、入力されたビデオ信号からシーンチェンジなどの特徴点を抽出する。

【００２５】AVエンコーダ１５は、入力されたビデオ信号とオーディオ信号を、それぞれ符号化し、符号化ビデオストリーム(V)、符号化オーディオストリーム(A)、およびAV同期等のシステム情報(S)をマルチプレクサ１６に出力する。

【００２６】符号化ビデオストリームは、例えば、MPEG（Moving Picture Expert Group）２方式により符号化されたビデオストリームであり、符号化オーディオストリームは、例えば、MPEG１方式により符号化されたオーディオストリームや、ドルビーAC3方式により符号化されたオーディオストリーム等である。マルチプレクサ１６は、入力されたビデオおよびオーディオのストリームを、入力システム情報に基づいて多重化して、スイッチ１７を介して多重化ストリーム解析部１８とソースパケッタイザ１９に出力する。

【００２７】多重化ストリームは、例えば、MPEG2トランスポートストリームやMPEG2プログラムストリームである。ソースパケッタイザ１９は、入力された多重化ストリームを、そのストリームを記録させる記録媒体１００のアプリケーションフォーマットに従って、ソースパケットから構成されるAVストリームを符号化する。AVストリームは、ECC（誤り訂正）符号化部２０、変調部２１で所定の処理が施され、書き込み部２２に出力される。書き込み部２２は、制御部２３から出力される制御信号に基づいて、記録媒体１００にAVストリームファイルを書き込む（記録する）。

【００２８】デジタルインタフェースまたはデジタルテレビジョンチューナから入力されるデジタルテレビジョン放送等のトランスポートストリームは、端子１３に入力される。端子１３に入力されたトランスポートストリームの記録方式には、２通りあり、それらは、トランスペアレントに記録する方式と、記録ビットレートを下げるなどの目的のために再エンコードをした後に記録する方式である。記録方式の指示情報は、ユーザインタフェースとしての端子２４から制御部２３へ入力される。

【００２９】入力トランスポートストリームをトランスペアレントに記録する場合、端子１３に入力されたトランスポートストリームは、多重化ストリーム解析部１８と、ソースパケッタイザ１９に出力される。これ以降の記録媒体１００へAVストリームが記録されるまでの処理は、上述の入力オーディオ浸透とビデオ信号を符号化して記録する場合と同一の処理なので、その説明は省略する。

【００３０】入力トランスポートストリームを再エンコードした後に記録する場合、端子１３に入力されたトランスポートストリームは、デマルチプレクサ２６に入力される。デマルチプレクサ２６は、入力されたトランスポートストリームに対してデマルチプレクス処理を施し、ビデオストリーム(V)、オーディオストリーム(A)、およびシステム情報(S)を抽出する。

【００３１】デマルチプレクサ２６により抽出されたストリーム（情報）のうち、ビデオストリームはAVデコーダ２７に、オーディオストリームとシステム情報はマルチプレクサ１６に、それぞれ出力される。AVデコーダ２７は、入力されたビデオストリームを復号し、その再生ビデオ信号をAVエンコーダ１５に出力する。AVエンコーダ１５は、入力ビデオ信号を符号化し、符号化ビデオストリーム(V)をマルチプレクサ１６に出力する。

【００３２】一方、デマルチプレクサ２６から出力され、マルチプレクサ１６に入力されたオーディオストリームとシステム情報、および、AVエンコーダ１５から出力されたビデオストリームは、入力システム情報に基づいて、多重化されて、多重化ストリームとして多重化ストリーム解析部１８とソースパケットタイザ１９にスイッチ１７を介して出力される。これ以後の記録媒体１００へAVストリームが記録されるまでの処理は、上述の入力オーディオ信号とビデオ信号を符号化して記録する場合と同一の処理なので、その説明は省略する。

【００３３】本実施の形態の記録再生装置１は、AVストリームのファイルを記録媒体１００に記録すると共に、そのファイルを説明するアプリケーションデータベース情報も記録する。アプリケーションデータベース情報は、制御部２３により作成される。制御部２３への入力情報は、解析部１４からの動画像の特徴情報、多重化ストリーム解析部１８からのAVストリームの特徴情報、および端子２４から入力されるユーザからの指示情報である。

【００３４】解析部１４から供給される動画像の特徴情報は、入力動画像信号の中の特徴的な画像に関係する情報であり、例えば、プログラムの開始点、シーンチェンジ点、コマーシャル（ＣＭ）の開始・終了点などの指定情報（マーク）であり、また、その指定場所の画像のサムネイル画像の情報も含まれる。

【００３５】多重化ストリーム解析部１８からのAVストリームの特徴情報は、記録されるAVストリームの符号化情報に関係する情報であり、例えば、AVストリーム内のIピクチャのアドレス情報、AVストリームの符号化パラメータ、AVストリームの中の符号化パラメータの変化点情報、ビデオストリームの中の特徴的な画像に関係する情報（マーク）などである。

【００３６】端子２４からのユーザの指示情報は、AVストリームの中の、ユーザが指定した再生区間の指定情報、その再生区間の内容を説明するキャラクター文字、ユーザが好みのシーンにセットするブックマークやリジューム点の情報などである。

【００３７】制御部２３は、上記の入力情報に基づいて、AVストリームのデータベース(Clip)、AVストリームの再生区間(PlayItem)をグループ化したもの（PlayList）のデータベース、記録媒体１００の記録内容の管理情報(info.dvr)、およびサムネイル画像の情報を作成す

る。これらの情報から構成されるアプリケーションデータベース情報は、AVストリームと同様にして、ECC符号化部２０、変調部２１で処理されて、書き込み部２２へ入力される。書き込み部２２は、制御部２３から出力される制御信号に基づいて、記録媒体１００へデータベースファイルを記録する。

【００３８】上述したアプリケーションデータベース情報についての詳細は後述する。

【００３９】このようにして記録媒体１００に記録されたAVストリームファイル（画像データと音声データのファイル）と、アプリケーションデータベース情報が再生される場合、まず、制御部２３は、読み出し部２８に対して、記録媒体１００からアプリケーションデータベース情報を読み出すように指示する。そして、読み出し部２８は、記録媒体１００からアプリケーションデータベース情報を読み出し、そのアプリケーションデータベース情報は、復調部２９、ECC復号部３０の処理を経て、制御部２３へ入力される。

【００４０】制御部２３は、アプリケーションデータベース情報に基づいて、記録媒体１００に記録されているPlayListの一覧を端子２４のユーザインタフェースへ出力する。ユーザは、PlayListの一覧から再生したいPlayListを選択し、再生を指定されたPlayListに関する情報が制御部２３へ入力される。制御部２３は、そのPlayListの再生に必要なAVストリームファイルの読み出しを、読み出し部２８に指示する。読み出し部２８は、その指示に従い、記録媒体１００から対応するAVストリームを読み出し復調部２９に出力する。復調部２９に入力されたAVストリームは、所定の処理が施されることにより復調され、さらにECC復号部３０の処理を経て、ソースデパケッタイザ３１出力される。

【００４１】ソースデパケッタイザ３１は、記録媒体１００から読み出され、所定の処理が施されたアプリケーションフォーマットのAVストリームを、デマルチプレクサ２６に出力できるストリームに変換する。デマルチプレクサ２６は、制御部２３により指定されたAVストリームの再生区間(PlayItem)を構成するビデオストリーム(V)、オーディオストリーム(A)、およびAV同期等のシステム情報(S)を、AVデコーダ２７に出力する。AVデコーダ２７は、ビデオストリームとオーディオストリームを復号し、再生ビデオ信号と再生オーディオ信号を、それぞれ対応する端子３２と端子３３から出力する。

【００４２】また、ユーザインタフェースとしての端子２４から、ランダムアクセス再生や特殊再生を指示する情報が入力された場合、制御部２３は、AVストリームのデータベース(Clip)の内容に基づいて、記憶媒体１００からのAVストリームの読み出し位置を決定し、そのAVストリームの読み出しを、読み出し部２８に指示する。例えば、ユーザにより選択されたPlayListを、所定の時刻から再生する場合、制御部２３は、指定された時刻に最も近いタイムスタンプを持つIピクチャからのデータを読み出すように読み出し部２８に指示する。

【００４３】また、ユーザによって高速再生(Fast-forward playback)が指示された場合、制御部２３は、AVストリームのデータベース(Clip)に基づいて、AVストリームの中のI-ピクチャデータを順次連続して読み出すように読み出し部２８に指示する。

【００４４】読み出し部２８は、指定されたランダムアクセスポイントからAVストリームのデータを読み出し、読み出されたデータは、後段の各部の処理を経て再生される。

【００４５】次に、ユーザが、記録媒体１００に記録されているAVストリームの編集をする場合を説明する。ユーザが、記録媒体１００に記録されているAVストリームの再生区間を指定して新しい再生経路を作成したい場合、例えば、番組Ａという歌番組から歌手Ａの部分を再生し、その後続けて、番組Ｂという歌番組の歌手Ａの部分を再生したいといった再生経路を作成したい場合、ユーザインタフェースとしての端子２４から再生区間の開始点（イン点）と終了点（アウト点）の情報が制御部２３に入力される。制御部２３は、AVストリームの再生区間(PlayItem)をグループ化したもの（PlayList）のデータベースを作成する。

【００４６】ユーザが、記録媒体１００に記録されているAVストリームの一部を消去したい場合、ユーザインタフェースとしての端子２４から消去区間のイン点とアウト点の情報が制御部２３に入力される。制御部２３は、必要なAVストリーム部分だけを参照するようにPlayListのデータベースを変更する。また、AVストリームの不必要なストリーム部分を消去するように、書き込み部２２に指示する。

【００４７】ユーザが、記録媒体１００に記録されているAVストリームの再生区間を指定して新しい再生経路を作成したい場合であり、かつ、それぞれの再生区間をシームレスに接続したい場合について説明する。このような場合、制御部２３は、AVストリームの再生区間(PlayItem)をグループ化したもの（PlayList）のデータベースを作成し、さらに、再生区間の接続点付近のビデオストリームの部分的な再エンコードと再多重化を行う。

【００４８】まず、端子２４から再生区間のイン点のピクチャの情報と、アウト点のピクチャの情報が制御部２３へ入力される。制御部２３は、読み出し部２８にイン点側ピクチャとアウト点側のピクチャを再生するために必要なデータの読み出しを指示する。そして、読み出し部２８は、記録媒体１００からデータを読み出し、そのデータは、復調部２９、ECC復号部３０、ソースデパケッタイザ３１を経て、デマルチプレクサ２６に出力される。

【００４９】制御部２３は、デマルチプレクサ２６に入力されたデータを解析して、ビデオストリームの再エン

コード方法（picture_coding_typeの変更、再エンコードする符号化ビット量の割り当て）と、再多重化方式を決定し、その方式をAVエンコーダ１５とマルチプレクサ１６に供給する。

【００５０】次に、デマルチプレクサ２６は、入力されたストリームをビデオストリーム(V)、オーディオストリーム(A)、およびシステム情報(S)に分離する。ビデオストリームは、「AVデコーダ２７に入力されるデータ」と「マルチプレクサ１６に入力されるデータ」がある。前者のデータは、再エンコードするために必要なデータであり、これはAVデコーダ２７で復号され、復号されたピクチャはAVエンコーダ１５で再エンコードされて、ビデオストリームにされる。後者のデータは、再エンコードをしないで、オリジナルのストリームからコピーされるデータである。オーディオストリーム、システム情報については、直接、マルチプレクサ１６に入力される。

【００５１】マルチプレクサ１６は、制御部２３から入力された情報に基づいて、入力ストリームを多重化し、多重化ストリームを出力する。多重化ストリームは、ECC符号化部２０、変調部２１で処理されて、書き込み部２２に入力される。書き込み部２２は、制御部２３から供給される制御信号に基づいて、記録媒体１００にAVストリームを記録する。

【００５２】以下に、アプリケーションデータベース情報や、その情報に基づく再生、編集といった操作に関する説明をする。図２は、アプリケーションフォーマットの構造を説明する図である。アプリケーションフォーマットは、AVストリームの管理のためにPlayListとClipの２つのレイヤをもつ。Volume Informationは、ディスク内のすべてのClipとPlayListの管理をする。ここでは、１つのAVストリームとその付属情報のペアを１つのオブジェクトと考え、それをClipと称する。AVストリームファイルはClip AV stream fileと称し、その付属情報は、Clip Information fileと称する。

【００５３】１つのClip AV stream fileは、MPEG2トランスポートストリームをアプリケーションフォーマットによって規定される構造に配置したデータをストアする。一般的に、ファイルは、バイト列として扱われるが、Clip AV stream fileのコンテンツは、時間軸上に展開され、Clipの中のエントリーポイントは、主に時間ベースで指定される。所定のClipへのアクセスポイントのタイムスタンプが与えられた時、Clip Information fileは、Clip AV stream fileの中でデータの読み出しを開始すべきアドレス情報を見つけるために役立つ。

【００５４】PlayListについて、図３を参照して説明する。PlayListは、Clipの中からユーザが見たい再生区間を選択し、それを簡単に編集することができるようにするために設けられている。１つのPlayListは、Clipの中の再生区間の集まりである。所定のClipの中の１つの再生区間は、PlayItemと呼ばれ、それは、時間軸上のイン点（IN）とアウト点（OUT）の対で表される。従って、PlayListは、複数のPlayItemが集まることにより構成される。

【００５５】PlayListには、２つのタイプがある。１つは、Real PlayListであり、もう１つは、Virtual PlayListである。Real PlayListは、それが参照しているClipのストリーム部分を共有している。すなわち、Real PlayListは、それの参照しているClipのストリーム部分に相当するデータ容量をディスクの中で占め、Real PlayListが消去された場合、それが参照しているClipのストリーム部分もまたデータが消去される。

【００５６】Virtual PlayListは、Clipのデータを共有していない。従って、Virtual PlayListが変更または消去されたとしても、Clipの内容には何も変化が生じない。

【００５７】次に、Real PlayListの編集について説明する。図４（Ａ）は、Real PlayListのクリエイト(create：作成)に関する図であり、AVストリームが新しいClipとして記録される場合、そのClip全体を参照するReal PlayListが新たに作成される操作である。

【００５８】図４（Ｂ）は、Real PlayListのディバイド(divide：分割)に関する図であり、Real PlayListが所望な点で分けられて、２つのReal PlayListに分割される操作である。この分割という操作は、例えば、１つのPlayListにより管理される１つのクリップ内に、２つの番組が管理されているような場合に、ユーザが１つ１つの番組として登録（記録）し直したいといったようなときに行われる。この操作により、Clipの内容が変更される（Clip自体が分割される）ことはない。

【００５９】図４（Ｃ）は、Real PlayListのコンバイン(combine：結合)に関する図であり、２つのReal PlayListを結合して、１つの新しいReal PlayListにする操作である。この結合という操作は、例えば、ユーザが２つの番組を１つの番組として登録し直したいといったようなときに行われる。この操作により、Clipが変更される（Clip自体が１つにされる）ことはない。

【００６０】図５（Ａ）は、Real PlayList全体のデリート(delete：削除)に関する図であり、所定のReal PlayList全体を消去する操作がされた場合、削除されたReal PlayListが参照するClipの、対応するストリーム部分も削除される。

【００６１】図５（Ｂ）は、Real PlayListの部分的な削除に関する図であり、Real PlayListの所望な部分が削除された場合、対応するPlayItemが、必要なClipのストリーム部分だけを参照するように変更される。そして、Clipの対応するストリーム部分は削除される。

【００６２】図５（Ｃ）は、Real PlayListのミニマイズ(Minimize：最小化)に関する図であり、Real PlayListに対応するPlayItemを、Virtual PlayListに必要なClipのストリーム部分だけを参照するようにする操作であ

る。Virtual PlayList にとって不必要なClipの、対応するストリーム部分は削除される。

【００６３】上述したような操作により、Real PlayListが変更されて、そのReal PlayListが参照するClipのストリーム部分が削除された場合、その削除されたClipを使用しているVirtual PlayListが存在し、そのVirtual PlayListにおいて、削除されたClipにより問題が生じる可能性がある。

【００６４】そのようなことが生じないように、ユーザに、削除という操作に対して、「そのReal PlayListが参照しているClipのストリーム部分を参照しているVirtual PlayListが存在し、もし、そのReal PlayListが消去されると、そのVirtual PlayListもまた消去されることになるが、それでも良いか？」といったメッセージなどを表示させることにより、確認（警告）を促した後に、ユーザの指示により削除の処理を実行、または、キャンセルする。または、Virtual PlayListを削除する代わりに、Real PlayListに対してミニマイズの操作が行われるようにする。

【００６５】次にVirtual PlayListに対する操作について説明する。Virtual PlayListに対して操作が行われたとしても、Clipの内容が変更されることはない。図６は、アセンブル(Assemble) 編集 (IN-OUT 編集)に関する図であり、ユーザが見たいと所望した再生区間のPlayItemを作り、Virtual PlayListを作成するといった操作である。PlayItem間のシームレス接続が、アプリケーションフォーマットによりサポートされている（後述）。

【００６６】図６（Ａ）に示したように、２つのReal PlayList１，２と、それぞれのRealPlayListに対応するClip１，２が存在している場合に、ユーザがReal PlayList１内の所定の区間（In１乃至Out１までの区間：PlayItem１）を再生区間として指示し、続けて再生する区間として、Real PlayList２内の所定の区間（In２乃至Out２までの区間：PlayItem２）を再生区間として指示したとき、図６（Ｂ）に示すように、PlayItem１とPlayItem２から構成される１つのVirtual PlayListが作成される。

【００６７】次に、Virtual PlayList の再編集(Re-editing)について説明する。再編集には、Virtual PlayListの中のイン点やアウト点の変更、Virtual PlayListへの新しいPlayItemの挿入(insert)や追加(append)、Virtual PlayListの中のPlayItemの削除などがある。また、Virtual PlayListそのものを削除することもできる。

【００６８】図７は、Virtual PlayListへのオーディオのアフレコ(Audio dubbing (post recording))に関する図であり、Virtual PlayListへのオーディオのアフレコをサブパスとして登録する操作のことである。このオーディオのアフレコは、アプリケーションフォーマットによりサポートされている。Virtual PlayListのメインパスのAVストリームに、付加的なオーディオストリームが、サブパスとして付加される。

【００６９】Real PlayListとVirtual PlayListで共通の操作として、図８に示すようなPlayListの再生順序の変更(Moving)がある。この操作は、ディスク(ボリューム)の中でのPlayListの再生順序の変更であり、アプリケーションフォーマットにおいて定義されるTable Of PlayList（図２０などを参照して後述する）によってサポートされる。この操作により、Clipの内容が変更されるようなことはない。

【００７０】次に、マーク（Mark）について説明する。マークは、ClipおよびPlayListの中のハイライトや特徴的な時間を指定するために設けられている。Clipに付加されるマークは、AVストリームの内容に起因する特徴的なシーンを指定する、例えば、シーンチェンジ点などである。PlayListを再生する時、そのPlayListが参照するClipのマークを参照して、使用する事ができる。

【００７１】PlayListに付加されるマークは、主にユーザによってセットされる、例えば、ブックマークやリジューム点などである。ClipまたはPlayListにマークをセットすることは、マークの時刻を示すタイムスタンプをマークリストに追加することにより行われる。また、マークを削除することは、マークリストの中から、そのマークのタイムスタンプを除去する事である。従って、マークの設定や削除により、AVストリームは何の変更もされない。

【００７２】次にサムネイルについて説明する。サムネイルは、Volume、PlayList、およびClipに付加される静止画である。サムネイルには、２つの種類があり、１つは、内容を表す代表画としてのサムネイルである。これは主としてユーザがカーソル（不図示）などを操作して見たいものを選択するためのメニュー画面で使われるものである。もう１つは、マークが指しているシーンを表す画像である。

【００７３】Volumeと各Playlistは代表画を持つことができるようにする必要がある。Volumeの代表画は、ディスク（記録媒体１００、以下、記録媒体１００はディスク状のものであるとし、適宜、ディスクと記述する）を記録再生装置１の所定の場所にセットした時に、そのディスクの内容を表す静止画を最初に表示する場合などに用いられることを想定している。Playlistの代表画は、Playlistを選択するメニュー画面において、Playlistの内容を表すための静止画として用いられることを想定している。

【００７４】Playlistの代表画として、Playlistの最初の画像をサムネイル（代表画）にすることが考えられるが、必ずしも再生時刻０の先頭の画像が内容を表す上で最適な画像とは限らない。そこで、Playlistのサムネイルとして、任意の画像をユーザが設定できるようにする。以上２種類のサムネイルをメニューサムネイルと称する。メニューサムネイルは頻繁に表示されるため、デ

ィスクから高速に読み出される必要がある。このため、すべてのメニューサムネイルを１つのファイルに格納することが効率的である。メニューサムネイルは、必ずしもボリューム内の動画から抜き出したピクチャである必要はなく、図１０に示すように、パーソナルコンピュータやデジタルスチルカメラから取り込まれた画像でもよい。

【００７５】一方、ClipとPlaylistには、複数個のマークを打てる必要があり、マーク位置の内容を知るためにマーク点の画像を容易に見ることが出来るようにする必要がある。このようなマーク点を表すピクチャをマークサムネイル（Mark Thumbnails）と称する。従って、サムネイルの元となる画像は、外部から取り込んだ画像よりも、マーク点の画像を抜き出したものが主となる。

【００７６】図１１は、PlayListに付けられるマークと、そのマークサムネイルの関係について示す図であり、図１２は、Clipに付けられるマークと、そのマークサムネイルの関係について示す図である。マークサムネイルは、メニューサムネイルと異なり、Playlistの詳細を表す時に、サブメニュー等で使われるため、短いアクセス時間で読み出されるようなことは要求されない。そのため、サムネイルが必要になる度に、記録再生装置１がファイルを開き、そのファイルの一部を読み出すことで多少時間がかかっても、問題にはならない。

【００７７】また、ボリューム内に存在するファイル数を減らすために、すべてのマークサムネイルは１つのファイルに格納するのがよい。Playlistはメニューサムネイル１つと複数のマークサムネイルを有することができるが、Clipは直接ユーザが選択する必要性がない（通常、Playlist経由で指定する）ため、メニューサムネイルを設ける必要はない。

【００７８】図１３は、上述したことを考慮した場合のメニューサムネイル、マークサムネイル、PlayList、およびClipの関係について示した図である。メニューサムネイルファイルには、PlayList毎に設けられたメニューサムネイルがファイルされている。メニューサムネイルファイルには、ディスクに記録されているデータの内容を代表するボリュームサムネイルが含まれている。マークサムネイルファイルは、各PlayList毎と各Clip毎に作成されたサムネイルがファイルされている。

【００７９】次に、CPI（Characteristic Point Information）について説明する。CPIは、Clipインフォメーションファイルに含まれるデータであり、主に、それはClipへのアクセスポイントのタイムスタンプが与えられた時、Clip AV stream fileの中でデータの読み出しを開始すべきデータアドレスを見つけるために用いられる。本実施の形態では、２種類のCPIを用いる。１つは、EP_mapであり、もう一つは、TU_mapである。

【００８０】EP_mapは、エントリーポイント(EP)データのリストであり、それはエレメンタリストリームおよびトランスポートストリームから抽出されたものである。これは、AVストリームの中でデコードを開始すべきエントリーポイントの場所を見つけるためのアドレス情報を持つ。１つのEPデータは、プレゼンテーションタイムスタンプ（PTS）と、そのPTSに対応するアクセスユニットのAVストリームの中のデータアドレスの対で構成される。

【００８１】EP_mapは、主に２つの目的のために使用される。第１に、PlayListの中でプレゼンテーションタイムスタンプによって参照されるアクセスユニットのAVストリームの中のデータアドレスを見つけるために使用される。第２に、ファーストフォワード再生やファーストリバース再生のために使用される。記録再生装置１が、入力AVストリームを記録する場合、そのストリームのシンタクスを解析することができるとき、EP_mapが作成され、ディスクに記録される。

【００８２】TU_mapは、デジタルインタフェースを通して入力されるトランスポートパケットの到着時刻に基づいたタイムユニット（TU）データのリストを持つ。これは、到着時刻ベースの時間とAVストリームの中のデータアドレスとの関係を与える。記録再生装置１が、入力AVストリームを記録する場合、そのストリームのシンタクスを解析することができないとき、TU_mapが作成され、ディスクに記録される。

【００８３】本実施の形態では、セルフエンコードのストリームフォーマット（SESF）を定義する。SESFは、アナログ入力信号を符号化する目的、およびデジタル入力信号（例えばDV）をデコードしてからMPEG2トランスポートストリームに符号化する場合に用いられる。

【００８４】SESFは、MPEG-2トランスポートストリームおよびAVストリームについてのエレメンタリストリームの符号化制限を定義する。記録再生装置１が、SESFストリームをエンコードし、記録する場合、EP_mapが作成され、ディスクに記録される。

【００８５】デジタル放送のストリームは、次に示す方式のうちのいずれかが用いられて記録媒体１００に記録される。まず、デジタル放送のストリームをSESFストリームにトランスコーディングする。この場合、記録されたストリームは、SESFに準拠しなければならない。この場合、EP_mapが作成されて、ディスクに記録されなければならない。

【００８６】あるいは、デジタル放送ストリームを構成するエレメンタリストリームを新しいエレメンタリストリームにトランスコーディングし、そのデジタル放送ストリームの規格化組織が定めるストリームフォーマットに準拠した新しいトランスポートストリームに再多重化する。この場合、EP_mapが作成されて、ディスクに記録されなければならない。

【００８７】例えば、入力ストリームがISDB（日本のデジタルBS放送の規格名称）準拠のMPEG-2トランスポート

ストリームであり、それがHDTVビデオストリームとMPEG AACオーディオストリームを含むとする。HDTVビデオストリームをSDTVビデオストリームにトランスコーディングし、そのSDTVビデオストリームとオリジナルのAACオーディオストリームをTSに再多重化する。SDTVストリームと記録されるトランスポートストリームは、共にISDBフォーマットに準拠しなければならない。

【００８８】デジタル放送のストリームが、記録媒体１００に記録される際の他の方式として、入力トランスポートストリームをトランスペアレントに記録する（入力トランスポートストリームを何も変更しないで記録する）場合であり、その時にEP_mapが作成されてディスクに記録される。

【００８９】または、入力トランスポートストリームをトランスペアレントに記録する（入力トランスポートストリームを何も変更しないで記録する）場合であり、その時にTU_mapが作成されてディスクに記録される。

【００９０】次にディレクトリとファイルについて説明する。以下、記録再生装置１をDVR（Digital Video Recording）と適宜記述する。図１４はディスク上のディレクトリ構造の一例を示す図である。DVRのディスク上に必要なディレクトリは、図１４に示したように、"DVR"ディレクトリを含むrootディレクトリ、"PLAYLIST"ディレクトリ、"CLIPINF"ディレクトリ、"M2TS"ディレクトリ、および"DATA"ディレクトリを含む"DVR"ディレクトリである。rootディレクトリの下に、これら以外のディレクトリを作成されるようにしても良いが、それらは、本実施の形態のアプリケーションフォーマットでは、無視されるとする。

【００９１】"DVR"ディレクトリの下には、DVRアプリケーションフォーマットによって規定される全てのファイルとディレクトリがストアされる。"DVR"ディレクトリは、４個のディレクトリを含む。"PLAYLIST"ディレクトリの下には、Real PlayListとVirtual PlayListのデータベースファイルが置かれる。このディレクトリは、PlayListが１つもなくても存在する。

【００９２】"CLIPINF"ディレクトリの下には、Clipのデータベースが置かれる。このディレクトリも、Clipが１つもなくても存在する。"M2TS"ディレクトリの下には、AVストリームファイルが置かれる。このディレクトリは、AVストリームファイルが１つもなくても存在する。"DATA"ディレクトリは、デジタルTV放送などのデータ放送のファイルがストアされる。

【００９３】"DVR"ディレクトリは、次に示すファイルをストアする。"info.dvr"ファイルは、DVRディレクトリの下に作られ、アプリケーションレイヤの全体的な情報をストアする。DVRディレクトリの下には、ただ一つのinfo.dvrがなければならない。ファイル名は、info.dvrに固定されるとする。"menu.thmb"ファイルは、メニューサムネイル画像に関連する情報をストアする。DVRディレクトリの下には、ゼロまたは１つのメニューサムネイルがなければならない。ファイル名は、memu.thmbに固定されるとする。メニューサムネイル画像が１つもない場合、このファイルは、存在しなくても良い。

【００９４】"mark.thmb"ファイルは、マークサムネイル画像に関連する情報をストアする。DVRディレクトリの下には、ゼロまたは１つのマークサムネイルがなければならない。ファイル名は、mark.thmbに固定されるとする。メニューサムネイル画像が１つもない場合、このファイルは、存在しなくても良い。

【００９５】"PLAYLIST"ディレクトリは、２種類のPlayListファイルをストアするものであり、それらは、Real PlayListとVirtual PlayListである。" xxxxx.rpls" ファイルは、１つのReal PlayListに関連する情報をストアする。それぞれのReal PlayList毎に、１つのファイルが作られる。ファイル名は、"xxxxx.rpls"である。ここで、"xxxxx"は、５個の０乃至９まで数字である。ファイル拡張子は、"rpls"でなければならないとする。

【００９６】"yyyyy.vpls"ファイルは、１つのVirtual PlayListに関連する情報をストアする。それぞれのVirtual PlayList毎に、１つのファイルが作られる。ファイル名は、"yyyyy.vpls"である。ここで、"yyyyy"は、５個の０乃至９まで数字である。ファイル拡張子は、"vpls"でなければならないとする。

【００９７】"CLIPINF"ディレクトリは、それぞれのAVストリームファイルに対応して、１つのファイルをストアする。"zzzzz.clpi" ファイルは、１つのAVストリームファイル(Clip AV stream file または Bridge-Clip AV stream file)に対応するClip Information fileである。ファイル名は、"zzzzz.clpi"であり、"zzzzz"は、５個の０乃至９までの数字である。ファイル拡張子は、"clpi"でなければならないとする。

【００９８】"M2TS"ディレクトリは、AVストリームのファイルをストアする。"zzzzz.m2ts"ファイルは、DVRシステムにより扱われるAVストリームファイルである。これは、Clip AV stream fileまたはBridge-Clip AV streamである。ファイル名は、"zzzzz.m2ts"であり、"zzzzz"は、５個の０乃至９までの数字である。ファイル拡張子は、"m2ts"でなければならないとする。

【００９９】" DATA" ディレクトリは、データ放送から伝送されるデータをストアするものであり、データとは、例えば、XML fileやMHEGファイルなどである。

【０１００】次に、各ディレクトリ（ファイル）のシンタクスとセマンティクスを説明する。まず、" info.dvr" ファイルについて説明する。図１５は、" info.dvr" ファイルのシンタクスを示す図である。" info.dvr" ファイルは、３個のオブジェクトから構成され、それらは、DVRVolume()、TableOfPlayLists()、およびMakerPrivateData()である。

【０１０１】図１５に示したinfo.dvrのシンタクスにつ

いて説明するに、TableOfPlayLists_Start_addressは、info.dvrファイルの先頭のバイトからの相対バイト数を単位として、TableOfPlayList()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

【0102】MakerPrivateData_Start_addressは、info.dvrファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。padding_word（パディングワード）は、info.dvrのシンタクスに従って挿入される。N1とN2は、ゼロまたは任意の正の整数である。それぞれのパディングワードは、任意の値を取るようにしても良い。

【0103】DVRVolume()は、ボリューム（ディスク）の内容を記述する情報をストアする。図16は、DVRVolume()のシンタクスを示す図である。図16に示したDVR Volume()のシンタクスを説明するに、version_numberは、このDVRVolume()のバージョンナンバーを示す4個のキャラクター文字を示す。version_numberは、ISO 646に従って、"0045"と符号化される。

【0104】lengthは、このlengthフィールドの直後からDVRVolume()の最後までのDVRVolume()のバイト数を示す32ビットの符号なし整数で表される。

【0105】ResumeVolume()は、ボリュームの中で最後に再生したReal PlayListまたはVirtual PlayListのファイル名を記憶している。ただし、Real PlayListまたはVirtual PlayListの再生をユーザが中断した時の再生位置は、PlayListMark()において定義されるresume-markにストアされる。

【0106】図17は、ResumeVolume()のシンタクスを示す図である。図17に示したResumeVolume()のシンタクスを説明するに、valid_flagは、この1ビットのフラグが1にセットされている場合、resume_PlayList_nameフィールドが有効であることを示し、このフラグが0にセットされている場合、resume_PlayList_nameフィールドが無効であることを示す。

【0107】resume_PlayList_nameの10バイトのフィールドは、リジュームされるべきReal PlayListまたはVirtual PlayListのファイル名を示す。

【0108】図16に示したDVRVolume()のシンタクスのなかの、UIAppInfoVolume は、ボリュームについてのユーザインタフェースアプリケーションのパラメータをストアする。図18は、UIAppInfoVolumeのシンタクスを示す図であり、そのセマンティクスを説明するに、character_setの8ビットのフィールドは、Volume_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図19に示される値に対応する。

【0109】name_lengthの8ビットフィールドは、Volume_nameフィールドの中に示されるボリューム名のバイト長を示す。Volume_nameのフィールドは、ボリュームの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクター文字であり、それはボリュームの名称を示す。Volume_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どんな値が入っていても良い。

【0110】Volume_protect_flagは、ボリュームの中のコンテンツを、ユーザに制限することなしに見せてよいかどうかを示すフラグである。このフラグが1にセットされている場合、ユーザが正しくPIN番号（パスワード）を入力できたときだけ、そのボリュームのコンテンツを、ユーザに見せる事（再生される事）が許可される。このフラグが0にセットされている場合、ユーザがPIN番号を入力しなくても、そのボリュームのコンテンツを、ユーザに見せる事が許可される。

【0111】最初に、ユーザが、ディスクをプレーヤへ挿入した時点において、もしこのフラグが0にセットされているか、または、このフラグが1にセットされていてもユーザがPIN番号を正しく入力できたならば、記録再生装置1は、そのディスクの中のPlayListの一覧を表示させる。それぞれのPlayListの再生制限は、volume_protect_flagとは無関係であり、それはUIAppInfoPlayList()の中に定義されるplayback_control_flagによって示される。

【0112】PINは、4個の0乃至9までの数字で構成され、それぞれの数字は、ISO/IEC 646に従って符号化される。ref_thumbnail_indexのフィールドは、ボリュームに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのボリュームにはサムネイル画像が付加されており、そのサムネイル画像は、menu.thumファイルの中にストアされている。その画像は、menu.thumファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、0xFFFF である場合、そのボリュームにはサムネイル画像が付加されていないことを示す。

【0113】次に図15に示したinfo.dvrのシンタクス内のTableOfPlayLists()について説明する。TableOfPlayLists()は、PlayList(Real PlayListとVirtual PlayList)のファイル名をストアする。ボリュームに記録されているすべてのPlayListファイルは、TableOfPlayList()の中に含まれる。TableOfPlayLists()は、ボリュームの中のPlayListのデフォルトの再生順序を示す。

【0114】図20は、TableOfPlayLists()のシンタクスを示す図であり、そのシンタクスについて説明するに、TableOfPlayListsのversion_numberは、このTableOfPlayListsのバージョンナンバーを示す4個のキャラクター文字を示す。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

【0115】lengthは、このlengthフィールドの直後からTableOfPlayLists()の最後までのTableOfPlayLists()

のバイト数を示す３２ビットの符号なしの整数である。number_of_PlayListsの１６ビットのフィールドは、PlayList_file_nameを含むfor-loopのループ回数を示す。この数字は、ボリュームに記録されているPlayListの数に等しくなければならない。PlayList_file_nameの１０バイトの数字は、PlayListのファイル名を示す。

【０１１６】図２１は、TableOfPlayLists()のシンタクスを別実施の構成を示す図である。図２１に示したシンタクスは、図２０に示したシンタクスに、UIAppinfoPlayList（後述）を含ませた構成とされている。このように、UIAppinfoPlayListを含ませた構成とすることで、TableOfPlayListsを読み出すだけで、メニュー画面を作成することが可能となる。ここでは、図２０に示したシンタクスを用いるとして以下の説明をする。

【０１１７】図１５に示したinfo.dvrのシンタクス内のMakersPrivateDataについて説明する。MakersPrivateDataは、記録再生装置１のメーカが、各社の特別なアプリケーションのために、MakersPrivateData()の中にメーカのプライベートデータを挿入できるように設けられている。各メーカのプライベートデータは、それを定義したメーカを識別するために標準化されたmaker_IDを持つ。MakersPrivateData()は、１つ以上のmaker_IDを含んでも良い。

【０１１８】所定のメーカが、プライベートデータを挿入したい時に、すでに他のメーカのプライベートデータがMakersPrivateData()に含まれていた場合、他のメーカは、既にある古いプライベートデータを消去するのではなく、新しいプライベートデータをMakersPrivateData()の中に追加するようにする。このように、本実施の形態においては、複数のメーカのプライベートデータが、１つのMakersPrivateData()に含まれることが可能であるようにする。

【０１１９】図２２は、MakersPrivateDataのシンタクスを示す図である。図２２に示したMakersPrivateDataのシンタクスについて説明するに、version_numberは、このMakersPrivateData()のバージョンナンバーを示す４個のキャラクター文字を示す。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からMakersPrivateData()の最後までのMakersPrivateData()のバイト数を示す３２ビットの符号なし整数を示す。

【０１２０】mpd_blocks_start_addressは、MakersPrivateData()の先頭のバイトからの相対バイト数を単位として、最初のmpd_block()の先頭バイトアドレスを示す。相対バイト数はゼロからカウントされる。number_of_maker_entriesは、MakersPrivateData()の中に含まれているメーカプライベートデータのエントリー数を与える１６ビットの符号なし整数である。MakersPrivateData()の中に、同じmaker_IDの値を持つメーカプライベートデータが２個以上存在してはならない。

【０１２１】mpd_block_sizeは、１０２４バイトを単位として、１つのmpd_blockの大きさを与える１６ビットの符号なし整数である。例えば、mpd_block_size=１ならば、それは１つのmpd_blockの大きさが１０２４バイトであることを示す。number_of_mpd_blocksは、MakersPrivateData()の中に含まれるmpd_blockの数を与える１６ビットの符号なし整数である。maker_IDは、そのメーカプライベートデータを作成したDVRシステムの製造メーカを示す16ビットの符号なし整数である。maker_IDに符号化される値は、このDVRフォーマットのライセンサによって指定される。

【０１２２】maker_model_codeは、そのメーカプライベートデータを作成したDVRシステムのモデルナンバーコードを示す１６ビットの符号なし整数である。maker_model_codeに符号化される値は、このフォーマットのライセンスを受けた製造メーカによって設定される。start_mpd_block_numberは、そのメーカプライベートデータが開始されるmpd_blockの番号を示す１６ビットの符号なし整数である。メーカプライベートデータの先頭データは、mpd_blockの先頭にアラインされなければならない。start_mpd_block_numberは、mpd_blockのfor-loopの中の変数jに対応する。

【０１２３】mpd_lengthは、バイト単位でメーカプライベートデータの大きさを示す３２ビットの符号なし整数である。mpd_blockは、メーカプライベートデータがストアされる領域である。MakersPrivateData()の中のすべてのmpd_blockは、同じサイズでなければならない。

【０１２４】次に、Real PlayList fileとVirtual PlayList fileについて、換言すれば、xxxxx.rplsとyyyyy.vplsについて説明する。図２３は、xxxxx.rpls（Real PlayList）、または、yyyyy.vpls（Virtual PlayList）のシンタクスを示す図である。xxxxx.rplsとyyyyy.vplsは、同一のシンタクス構成をもつ。xxxxx.rplsとyyyyy.vplsは、それぞれ、３個のオブジェクトから構成され、それらは、PlayList()、PlayListMark()、およびMakerPrivateData()である。

【０１２５】PlayListMark_Start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、PlayListMark()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

【０１２６】MakerPrivateData_Start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

【０１２７】padding_word（パディングワード）は、PlayListファイルのシンタクスにしたがって挿入され、Ｎ１とＮ２は、ゼロまたは任意の正の整数である。それぞれのパディングワードは、任意の値を取るようにしても良い。

【０１２８】ここで、既に、簡便に説明したが、PlayLi

stについてさらに説明する。ディスク内にあるすべてのReal PlayListによって、Bridge-Clip（後述）を除くすべてのClipの中の再生区間が参照されていなければならない。かつ、2つ以上のRealPlayListが、それらのPlayItemで示される再生区間を同一のClipの中でオーバーラップさせてはならない。

【0129】図24を参照してさらに説明するに、図24（A）に示したように、全てのClipは、対応するReal PlayListが存在する。この規則は、図24（B）に示したように、編集作業が行われた後においても守られる。従って、全てのClipは、どれかしらのReal PlayListを参照することにより、必ず視聴することが可能である。

【0130】図24（C）に示したように、Virtual PlayListの再生区間は、Real PlayListの再生区間またはBridge-Clipの再生区間の中に含まれていなければならない。どのVirtual PlayListにも参照されないBridge-Clipがディスクの中に存在してはならない。

【0131】Real PlayListは、PlayItemのリストを含むが、SubPlayItemを含んではならない。Virtual PlayListは、PlayItemのリストを含み、PlayList()の中に示されるCPI_typeがEP_map typeであり、かつPlayList_typeが0（ビデオとオーディオを含むPlayList）である場合、Virtual PlayListは、ひとつのSubPlayItemを含む事ができる。本実施の形態におけるPlayList()では、SubPlayIteはオーディオのアフレコの目的にだけに使用される、そして、1つのVirtual PlayListが持つSubPlayItemの数は、0または1でなければならない。

【0132】次に、PlayListについて説明する。図25は、PlayListのシンタクスを示す図である。図25に示したPlayListのシンタクスを説明するに、version_numberは、このPlayList()のバージョンナンバーを示す4個のキャラクター文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からPlayList()の最後までのPlayList()のバイト数を示す32ビットの符号なし整数である。PlayList_typeは、このPlayListのタイプを示す8ビットのフィールドであり、その一例を図26に示す。

【0133】CPI_typeは、1ビットのフラグであり、PlayItem()およびSubPlayItem()によって参照されるClipのCPI_typeの値を示す。1つのPlayListによって参照される全てのClipは、それらのCPI()の中に定義されるCPI_typeの値が同じでなければならない。number_of_PlayItemsは、PlayListの中にあるPlayItemの数を示す16ビットのフィールドである。

【0134】所定のPlayItem()に対応するPlayItem_idは、PlayItem()を含むfor-loopの中で、そのPlayItem()の現れる順番により定義される。PlayItem_idは、0から開始される。number_of_SubPlayItemsは、PlayListの中にあるSubPlayItemの数を示す16ビットのフィールドである。この値は、0または1である。付加的なオーディオストリームのパス(オーディオストリームパス)は、サブパスの一種である。

【0135】次に、図25に示したPlayListのシンタクスのUIAppInfoPlayListについて説明する。UIAppInfoPlayListは、PlayListについてのユーザインタフェースアプリケーションのパラメータをストアする。図27は、UIAppInfoPlayListのシンタクスを示す図である。図27に示したUIAppInfoPlayListのシンタクスを説明するに、character_setは、8ビットのフィールドであり、PlayList_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図19に示したテーブルに準拠する値に対応する。

【0136】name_lengthは、8ビットフィールドであり、PlayList_nameフィールドの中に示されるPlayList名のバイト長を示す。PlayList_nameのフィールドは、PlayListの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクター文字であり、それはPlayListの名称を示す。PlayList_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どんな値が入っていても良い。

【0137】record_time_and_dateは、PlayListが記録された時の日時をストアする56ビットのフィールドである。このフィールドは、年／月／日／時／分／秒について、14個の数字を4ビットのBinary Coded Decimal (BCD)で符号化したものである。例えば、2001/12/23:01:02:03 は、"0x20011223010203"と符号化される。

【0138】durationは、PlayListの総再生時間を時間／分／秒の単位で示した24ビットのフィールドである。このフィールドは、6個の数字を4ビットのBinary CodedDecimal(BCD)で符号化したものである。例えば、01:45:30は、"0x014530"と符号化される。

【0139】valid_periodは、PlayListが有効である期間を示す32ビットのフィールドである。このフィールドは、8個の数字を4ビットのBinary Coded Decimal(BCD)で符号化したものである。例えば、記録再生装置1は、この有効期間の過ぎたPlayListを自動消去する、といったように用いられる。例えば、2001/05/07 は、"0x20010507"と符号化される。

【0140】maker_idは、そのPlayListを最後に更新したDVRプレーヤ（記録再生装置1）の製造者を示す16ビットの符号なし整数である。maker_idに符号化される値は、DVRフォーマットのライセンサによって割り当てられる。maker_codeは、そのPlayListを最後に更新したDVRプレーヤのモデル番号を示す16ビットの符号なし整数である。maker_codeに符号化される値は、DVRフォーマットのライセンスを受けた製造者によって決められる。

【0141】playback_control_flagのフラグが1にセ

ットされている場合、ユーザが正しくPIN番号を入力できた場合にだけ、そのPlayListは再生される。このフラグが0にセットされている場合、ユーザがPIN番号を入力しなくても、ユーザは、そのPlayListを視聴することができる。

【0142】write_protect_flagは、図28（A）にテーブルを示すように、1にセットされている場合、write_protect_flagを除いて、そのPlayListの内容は、消去および変更されない。このフラグが0にセットされている場合、ユーザは、そのPlayListを自由に消去および変更できる。このフラグが1にセットされている場合、ユーザが、そのPlayListを消去、編集、または上書きする前に、記録再生装置1はユーザに再確認するようなメッセージを表示させる。

【0143】write_protect_flagが0にセットされているReal PlayListが存在し、かつ、そのReal PlayListのClipを参照するVirtual PlayListが存在し、そのVirtual PlayListのwrite_protect_flagが1にセットされていても良い。ユーザが、RealPlayListを消去しようとする場合、記録再生装置1は、そのReal PlayListを消去する前に、上記Virtual PlayListの存在をユーザに警告するか、または、そのReal PlayListを"Minimize"する。

【0144】is_played_flagは、図28（B）に示すように、フラグが1にセットされている場合、そのPlayListは、記録されてから一度は再生されたことを示し、0にセットされている場合、そのPlayListは、記録されてから一度も再生されたことがないことを示す。

【0145】archiveは、図28（C）に示すように、そのPlayListがオリジナルであるか、コピーされたものであるかを示す2ビットのフィールドである。ref_thumbnail_index のフィールドは、PlayListを代表するサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのPlayListには、PlayListを代表するサムネイル画像が付加されており、そのサムネイル画像は、menu.thum ファイルの中にストアされている。その画像は、menu.thumファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、0xFFFF である場合、そのPlayListには、PlayListを代表するサムネイル画像が付加されていない。

【0146】次にPlayItemについて説明する。1つのPlayItem()は、基本的に次のデータを含む。Clipのファイル名を指定するためのClip_information_file_name、Clipの再生区間を特定するためのIN_timeとOUT_timeのペア、PlayList()において定義されるCPI_typeがEP_map typeである場合、IN_timeとOUT_timeが参照するところのSTC_sequence_id、および、先行するPlayItemと現在のPlayItemとの接続の状態を示すところのconnection_conditionである。

【0147】PlayListが2つ以上のPlayItemから構成される時、それらのPlayItemはPlayListのグローバル時間軸上に、時間のギャップまたはオーバーラップなしに一列に並べられる。PlayList()において定義されるCPI_typeがEP_map typeであり、かつ現在のPlayItemがBridgeSequence()を持たない時、そのPlayItemにおいて定義されるIN_timeとOUT_timeのペアは、STC_sequence_idによって指定される同じSTC連続区間上の時間を指していなければならない。そのような例を図29に示す。

【0148】図30は、PlayList()において定義されるCPI_typeがEP_map typeであり、かつ現在のPlayItemがBridgeSequence()を持つ時、次に説明する規則が適用される場合を示している。現在のPlayItemに先行するPlayItemのIN_time（図の中でIN_time1と示されているもの）は、先行するPlayItemのSTC_sequence_idによって指定されるSTC連続区間上の時間を指している。先行するPlayItemのOUT_time（図の中でOUT_time1と示されているもの）は、現在のPlayItemのBridgeSequenceInfo()の中で指定されるBridge-Clipの中の時間を指している。このOUT_timeは、後述する符号化制限に従っていなければならない。

【0149】現在のPlayItemのIN_time（図の中でIN_time2と示されているもの）は、現在のPlayItemのBridgeSequenceInfo()の中で指定されるBridge-Clipの中の時間を指している。このIN_timeも、後述する符号化制限に従っていなければならない。現在のPlayItemのPlayItemのOUT_time（図の中でOUT_time2と示されているもの）は、現在のPlayItemのSTC_sequence_idによって指定されるSTC連続区間上の時間を指している。

【0150】図31に示すように、PlayList()のCPI_typeがTU_map typeである場合、PlayItemのIN_timeとOUT_timeのペアは、同じClip AVストリーム上の時間を指している。

【0151】PlayItemのシンタクスは、図32に示すようになる。図32に示したPlayItemのシンタクスを説明するに、Clip_Information_file_nameのフィールドは、ClipInformation fileのファイル名を示す。このClip Information fileのClipInfo()において定義されるClip_stream_typeは、Clip AV streamを示していなければならない。

【0152】STC_sequence_idは、8ビットのフィールドであり、PlayItemが参照するSTC連続区間のSTC_sequence_idを示す。PlayList()の中で指定されるCPI_typeがTU_map typeである場合、この8ビットフィールドは何も意味を持たず、0にセットされる。IN_timeは、32ビットフィールドであり、PlayItemの再生開始時刻をストアする。IN_timeのセマンティクスは、図33に示すように、PlayList()において定義されるCPI_typeによって異なる。

【0153】OUT_timeは、32ビットフィールドであり、PlayItemの再生終了時刻をストアする。OUT_timeの

セマンティクスは、図３４に示すように、PlayList()において定義されるCPI_typeによって異なる。

【０１５４】Connection_Conditionは、図３５に示したような先行するPlayItemと、現在のPlayItemとの間の接続状態を示す２ビットのフィールドである。図３６は、図３５に示したConnection_Conditionの各状態について説明する図である。

【０１５５】次に、BridgeSequenceInfoについて、図３７を参照して説明する。BridgeSequenceInfo()は、現在のPlayItemの付属情報であり、次に示す情報を持つ。Bridge-Clip AV streamファイルとそれに対応するClip Information fileを指定するBridge_Clip_Information_file_nameを含む。

【０１５６】また、先行するPlayItemが参照するClip AV stream上のソースパケットのアドレスであり、このソースパケットに続いてBridge-Clip AV streamファイルの最初のソースパケットが接続される。このアドレスは、RSPN_exit_from_previous_Clipと称される。さらに現在のPlayItemが参照するClip AV stream上のソースパケットのアドレスであり、このソースパケットの前にBridge-Clip AV streamファイルの最後のソースパケットが接続される。このアドレスは、RSPN_enter_to_current_Clipと称される。

【０１５７】図３７において、RSPN_arrival_time_discontinuityは、the Bridge-Clip AVstreamファイルの中でアライバルタイムベースの不連続点があるところのソースパケットのアドレスを示す。このアドレスは、ClipInfo()の中において定義される。

【０１５８】図３８は、BridgeSequenceinfoのシンタクスを示す図である。図３８に示したBridgeSequenceinfoのシンタクスを説明するに、Bridge_Clip_Information_file_nameのフィールドは、Bridge-Clip AV streamファイルに対応するClip Information fileのファイル名を示す。このClip Information fileのClipInfo()において定義されるClip_stream_typeは、'Bridge-Clip AV stream'を示していなければならない。

【０１５９】RSPN_exit_from_previous_Clipの３２ビットフィールドは、先行するPlayItemが参照するClip AV stream上のソースパケットの相対アドレスであり、このソースパケットに続いてBridge-Clip AV streamファイルの最初のソースパケットが接続される。RSPN_exit_from_previous_Clipは、ソースパケット番号を単位とする大きさであり、先行するPlayItemが参照するClip AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。

【０１６０】RSPN_enter_to_current_Clipの３２ビットフィールドは、現在のPlayItemが参照するClip AV stream上のソースパケットの相対アドレスであり、このソースパケットの前にBridge-Clip AV streamファイルの最後のソースパケットが接続される。RSPN_exit_from_previous_Clipは、ソースパケット番号を単位とする大きさであり、現在のPlayItemが参照するClip AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。

【０１６１】次に、SubPlayItemについて、図３９を参照して説明する。SubPlayItem()の使用は、PlayList()のCPI_typeがEP_map typeである場合だけに許される。本実施の形態においては、SubPlayItemはオーディオのアフレコの目的のためだけに使用されるとする。SubPlayItem()は、次に示すデータを含む。まず、PlayListの中のsub pathが参照するClipを指定するためのClip_information_file_ nameを含む。

【０１６２】また、Clipの中のsub pathの再生区間を指定するためのSubPath_IN_time と SubPath_OUT_timeを含む。さらに、main pathの時間軸上でsub pathが再生開始する時刻を指定するためのsync_PlayItem_id と sync_start_PTS_of_PlayItemを含む。sub pathに参照されるオーディオのClip AV streamは、STC不連続点（システムタイムベースの不連続点）を含んではならない。sub pathに使われるClipのオーディオサンプルのクロックは、main pathのオーディオサンプルのクロックにロックされている。

【０１６３】図４０は、SubPlayItemのシンタクスを示す図である。図４０に示したSubPlayItemのシンタクスを説明するに、Clip_Information_file_nameのフィールドは、Clip Information fileのファイル名を示し、それはPlayListの中でsub pathによって使用される。このClip Information fileのClipInfo()において定義されるClip_stream_typeは、Clip AV streamを示していなければならない。

【０１６４】SubPath_typeの８ビットのフィールドは、sub pathのタイプを示す。ここでは、図４１に示すように、'0x00'しか設定されておらず、他の値は、将来のために確保されている。

【０１６５】sync_PlayItem_idの８ビットのフィールドは、main pathの時間軸上でsub pathが再生開始する時刻が含まれるPlayItemのPlayItem_idを示す。所定のPlayItemに対応するPlayItem_idの値は、PlayList()において定義される（図２５参照）。

【０１６６】sync_start_PTS_of_PlayItemの３２ビットのフィールドは、main pathの時間軸上でsub pathが再生開始する時刻を示し、sync_PlayItem_idで参照されるPlayItem上のPTS(Presentaiotn Time Stamp)の上位３２ビットを示す。SubPath_IN_timeの３２ビットフィールドは、Sub pathの再生開始時刻をストアする。SubPath_IN_timeは、Sub Pathの中で最初のプレゼンテーションユニットに対応する３３ビット長のPTSの上位３２ビットを示す。

【0167】SubPath_OUT_timeの32ビットフィールドは、Sub pathの再生終了時刻をストアする。SubPath_OUT_timeは、次式によって算出されるPresenation_end_TSの値の上位32ビットを示す。

Presentation_end_TS = PTS_out + AU_duration

ここで、PTS_outは、SubPathの最後のプレゼンテーションユニットに対応する33ビット長のPTSである。AU_durationは、SubPathの最後のプレゼンテーションユニットの90kHz単位の表示期間である。

【0168】次に、図23に示したxxxxx.rplsとyyyyy.vplsのシンタクス内のPlayListMark()について説明する。PlayListについてのマーク情報は、このPlayListMarkにストアされる。図42は、PlayListMarkのシンタクスを示す図である。図42に示したPlayListMarkのシンタクスについて説明するに、version_numberは、このPlayListMark()のバージョンナンバーを示す4個のキャラクター文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

【0169】lengthは、このlengthフィールドの直後からPlayListMark()の最後までのPlayListMark()のバイト数を示す32ビットの符号なし整数である。number_of_PlayList_marksは、PlayListMarkの中にストアされているマークの個数を示す16ビットの符号なし整数である。number_of_PlayList_marks は、0であってもよい。mark_typeは、マークのタイプを示す8ビットのフィールドであり、図43に示すテーブルに従って符号化される。

【0170】mark_time_stampの32ビットフィールドは、マークが指定されたポイントを示すタイムスタンプをストアする。mark_time_stampのセマンティクスは、図44に示すように、PlayList()において定義されるCPI_typeによって異なる。PlayItem_idは、マークが置かれているところのPlayItemを指定する8ビットのフィールドである。所定のPlayItemに対応するPlayItem_idの値は、PlayList()において定義される（図25参照）。

【0171】character_setの8ビットのフィールドは、mark_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図19に示した値に対応する。name_lengthの8ビットフィールドは、Mark_nameフィールドの中に示されるマーク名のバイト長を示す。mark_nameのフィールドは、マークの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクター文字であり、それはマークの名称を示す。Mark_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どのような値が設定されても良い。

【0172】ref_thumbnail_indexのフィールドは、マークに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのマークにはサムネイル画像が付加されており、そのサムネイル画像は、mark.thmbファイルの中にストアされている。その画像は、mark.thmbファイルの中でref_thumbnail_indexの値を用いて参照される（後述）。ref_thumbnail_indexフィールドが、0xFFFF である場合、そのマークにはサムネイル画像が付加されていない事を示す。

【0173】次に、Clip information fileについて説明する。zzzzz.clpi（Clip information fileファイル）は、図45に示すように6個のオブジェクトから構成される。それらは、ClipInfo()、STC_Info()、ProgramInfo()、CPI()、ClipMark()、およびMakerPrivateData()である。AVストリーム(Clip AVストリームまたはBridge-Clip AV stream)とそれに対応するClip Informationファイルは、同じ数字列の"zzzzz"が使用される。

【0174】図45に示したzzzzz.clpi（Clip information fileファイル）のシンタクスについて説明するに、ClipInfo_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ClipInfo()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

【0175】STC_Info_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、STC_Info()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。ProgramInfo_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ProgramInfo()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。CPI_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、CPI()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

【0176】ClipMark_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ClipMark()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。MakerPrivateData_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData ()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。padding_word（パディングワード）は、zzzzz.clpiファイルのシンタクスにしたがって挿入される。N1，N2，N3，N4、およびN5は、ゼロまたは任意の正の整数でなければならない。それぞれのパディングワードは、任意の値がとられるようにしても良い。

【0177】次に、ClipInfoについて説明する。図46は、ClipInfoのシンタクスを示す図である。ClipInfo()は、それに対応するAVストリームファイル（Clip AVストリームまたは Bridge-Clip AVストリームファイル）の属性情報をストアする。

【0178】図46に示したClipInfoのシンタクスについて説明するに、version_numberは、このClipInfo()のバージョンナンバーを示す4個のキャラクター文字であ

る。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からClipInfo()の最後までのClipInfo()のバイト数を示す３２ビットの符号なし整数である。Clip_stream_typeの８ビットのフィールドは、図４７に示すように、Clip Informationファイルに対応するAVストリームのタイプを示す。それぞれのタイプのAVストリームのストリームタイプについては後述する。

【０１７９】offset_SPNの３２ビットのフィールドは、AVストリーム（Clip AVストリームまたは Bridge-Clip AVストリーム）ファイルの最初のソースパケットについてのソースパケット番号のオフセット値を与える。AVストリームファイルが最初にディスクに記録される時、このoffset_SPNは０でなければならない。

【０１８０】図４８に示すように、AVストリームファイルのはじめの部分が編集によって消去された時、offset_SPNは、ゼロ以外の値をとっても良い。本実施の形態では、offset_SPNを参照する相対ソースパケット番号（相対アドレス）が、しばしば、RSPN_xxx（xxxは変形する。例. RSPN_EP_start）の形式でシンタクスの中に記述されている。相対ソースパケット番号は、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからoffset_SPNの値を初期値としてカウントされる。

【０１８１】AVストリームファイルの最初のソースパケットから相対ソースパケットまでのソースパケットの数（SPN_xxx）は、次式で算出される。

SPN_xxx = RSPN_xxx - offset_SPN

図４８に、offset_SPN が、４である場合の例を示す。

【０１８２】TS_recording_rateは、２４ビットの符号なし整数であり、この値は、DVRドライブ（書き込み部２２）へまたはDVRドライブ（読み出し部２８）からのAVストリームの必要な入出力のビットレートを与える。record_time_and_dateは、Clipに対応するAVストリームが記録された時の日時をストアする５６ビットのフィールドであり、年／月／日／時／分／秒について、１４個の数字を４ビットのBinary Coded Decimal(BCD)で符号化したものである。例えば、2001/12/23:01:02:03は、0x20011223010203"と符号化される。

【０１８３】durationは、Clipの総再生時間をアライバルタイムクロックに基づいた時間／分／秒の単位で示した２４ビットのフィールドである。このフィールドは、６個の数字を４ビットのBinary Coded Decimal(BCD)で符号化したものである。例えば、01:45:30は、0x014530"と符号化される。

【０１８４】time_controlled_flag:のフラグは、AVストリームファイルの記録モードを示す。このtime_controlled_flagが１である場合、記録モードは、記録してからの時間経過に対してファイルサイズが比例するようにして記録されるモードであることを示し、次式に示す条件を満たさなければならない。

$$TS\_average\_rate * 192/188 * (t - start\_time) - \alpha <= size\_clip(t) <= TS\_average\_rate * 192/188 * (t - start\_time) + \alpha$$

ここで、TS_average_rateは、AVストリームファイルのトランスポートストリームの平均ビットレートをbytes/second の単位で表したものである。

【０１８５】また、上式において、tは、秒単位で表される時間を示し、start_timeは、AVストリームファイルの最初のソースパケットが記録された時の時刻であり、秒単位で表される。size_clip(t)は、 時刻ｔにおけるAVストリームファイルのサイズをバイト単位で表したものであり、例えば、start_timeから時刻tまでに１０個のソースパケットが記録された場合、size_clip(t)は10*192バイトである。αは、TS_average_rateに依存する定数である。

【０１８６】time_controlled_flagが0にセットされている場合、記録モードは、記録の時間経過とAVストリームのファイルサイズが比例するように制御していないことを示す。例えば、これは入力トランスポートストリームをトランスペアレント記録する場合である。

【０１８７】TS_average_rateは、time_controlled_flagが１にセットされている場合、この２４ビットのフィールドは、上式で用いているTS_average_rateの値を示す。time_controlled_flagが０にセットされている場合、このフィールドは、何も意味を持たず、０にセットされなければならない。例えば、可変ビットレートのトランスポートストリームは、次に示す手順により符号化される。まずトランスポートレートをTS_recording_rateの値にセットする。次に、ビデオストリームを可変ビットレートで符号化する。そして、ヌルパケットを使用しない事によって、間欠的にトランスポートパケットを符号化する。

【０１８８】RSPN_arrival_time_discontinuityの３２ビットフィールドは、Bridge-Clip AV streamファイル上でアライバルタイムベースの不連続が発生する場所の相対アドレスである。RSPN_arrival_time_discontinuityは、ソースパケット番号を単位とする大きさであり、Bridge-Clip AV streamファイルの最初のソースパケットからClipInfo() において定義されるoffset_SPNの値を初期値としてカウントされる。そのBridge-Clip AV streamファイルの中での絶対アドレスは、上述した

SPN_xxx = RSPN_xxx - offset_SPN

に基づいて算出される。

【０１８９】reserved_for_system_useの144ビットのフィールドは、システム用にリザーブされている。is_format_identifier_validのフラグが１である時、format_identifierのフィールドが有効であることを示す。is_or

iginal_network_ID_validのフラグが１である場合、original_network_IDのフィールドが有効であることを示す。is_transport_stream_ID_validのフラグが１である場合、transport_stream_IDのフィールドが有効であることを示す。is_servece_ID_validのフラグが１である場合、servece_IDのフィールドが有効であることを示す。

【０１９０】is_ country_code_validのフラグが１である時、country_codeのフィールドが有効であることを示す。format_identifierの32ビットフィールドは、トランスポートストリームの中でregistration deascriotor（ISO/IEC13818-1で定義されている）が持つformat_identifierの値を示す。original_network_IDの１６ビットフィールドは、トランスポートストリームの中で定義されているoriginal_network_IDの値を示す。transport_stream_IDの１６ビットフィールドは、トランスポートストリームの中で定義されているtransport_stream_IDの値を示す。

【０１９１】servece_IDの１６ビットフィールドは、トランスポートストリームの中で定義されているservece_IDの値を示す。country_codeの２４ビットのフィールドは、ISO3166によって定義されるカントリーコードを示す。それぞれのキャラクター文字は、ISO8859-1で符号化される。例えば、日本は"JPN"と表され、"0x4A 0x50 0x4E"と符号化される。stream_format_nameは、トランスポートストリームのストリーム定義をしているフォーマット機関の名称を示すISO-646の１６個のキャラクターコードである。このフィールドの中の無効なバイトは、値'0xFF'がセットされる。

【０１９２】format_identifier、original_network_ID、transport_stream_ID、 servece_ID,country_code、およびstream_format_nameは、トランスポートストリームのサービスプロバイダを示すものであり、これにより、オーディオやビデオストリームの符号化制限、SI(サービスインフォメーション)の規格やオーディオビデオストリーム以外のプライベートデータストリームのストリーム定義を認識することができる。これらの情報は、デコーダが、そのストリームをデコードできるか否か、そしてデコードできる場合にデコード開始前にデコーダシステムの初期設定を行うために用いることが可能である。

【０１９３】次に、STC_Infoについて説明する。ここでは、MPEG-2トランスポートストリームの中でSTCの不連続点（システムタイムベースの不連続点）を含まない時間区間をSTC_sequenceと称し、Clipの中で、STC_sequenceは、STC_sequence_idの値によって特定される。図５０は、連続なＳＴＣ区間について説明する図である。同じSTC_sequenceの中で同じSTCの値は、決して現れない（ただし、後述するように、Clipの最大時間長は制限されている）。従って、同じSTC_sequenceの中で同じPTSの値もまた、決して現れない。AVストリームが、N(N>0)個のSTC不連続点を含む場合、Clipのシステムタイムベースは、(N+1)個のSTC_sequenceに分割される。

【０１９４】STC_Infoは、STCの不連続（システムタイムベースの不連続）が発生する場所のアドレスをストアする。図５１を参照して説明するように、RSPN_STC_startが、そのアドレスを示し、最後のSTC_sequenceを除くk番目（k>=0）のSTC_sequenceは、k番目のRSPN_STC_startで参照されるソースパケットが到着した時刻から始まり、(k+1)番目のRSPN_STC_startで参照されるソースパケットが到着した時刻で終わる。最後のSTC_sequenceは、最後のRSPN_STC_startで参照されるソースパケットが到着した時刻から始まり、最後のソースパケットが到着した時刻で終了する。

【０１９５】図５２は、STC_Infoのシンタクスを示す図である。図５２に示したSTC_Infoのシンタクスについて説明するに、version_numberは、このSTC_Info()のバージョンナンバーを示す４個のキャラクター文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

【０１９６】lengthは、このlengthフィールドの直後からSTC_Info()の最後までのSTC_Info()のバイト数を示す３２ビットの符号なし整数である。CPI()のCPI_typeがTU_map typeを示す場合、このlengthフィールドはゼロをセットしても良い。CPI()のCPI_typeがEP_map typeを示す場合、num_of_STC_sequencesは1以上の値でなければならない。

【０１９７】num_of_STC_sequencesの８ビットの符号なし整数は、Clipの中でのSTC_sequenceの数を示す。この値は、このフィールドに続くfor-loopのループ回数を示す。所定のSTC_sequenceに対応するSTC_sequence_idは、RSPN_STC_startを含むfor-loopの中で、そのSTC_sequenceに対応するRSPN_STC_startの現れる順番により定義されるものである。STC_sequence_idは、０から開始される。

【０１９８】RSPN_STC_startの３２ビットフィールドは、AVストリームファイル上でSTC_sequenceが開始するアドレスを示す。RSPN_STC_startは、AVストリームファイルの中でシステムタイムベースの不連続点が発生するアドレスを示す。RSPN_STC_startは、AVストリームの中で新しいシステムタイムベースの最初のPCRを持つソースパケットの相対アドレスとしても良い。RSPN_STC_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAV streamファイルの中での絶対アドレスは、既に上述したSPN_xxx = RSPN_xxx - offset_SPNにより算出される。

【０１９９】次に、図４５に示したzzzzz.clipのシンタクス内のProgramInfoについて説明する。図５３を参照

しながら説明するに、ここでは、Clipの中で次の特徴をもつ時間区間をprogram_sequenceと呼ぶ。まず、PCR_PIDの値が変わらない。次に、ビデオエレメンタリストリームの数が変化しない。また、それぞれのビデオストリームについてのPIDの値とそのVideoCodingInfoによって定義される符号化情報が変化しない。さらに、オーディオエレメンタリストリームの数が変化しない。また、それぞれのオーディオストリームについてのPIDの値とそのAudioCodingInfoによって定義される符号化情報が変化しない。

【0200】program_sequenceは、同一の時刻において、ただ1つのシステムタイムベースを持つ。program_sequenceは、同一の時刻において、ただ1つのPMTを持つ。ProgramInfo()は、program_sequenceが開始する場所のアドレスをストアする。RSPN_program_sequence_startが、そのアドレスを示す。

【0201】図54は、ProgramInfoのシンタクスを示す図である。図54に示したProgramInfoのシンタクを説明するに、version_numberは、このProgramInfo()のバージョンナンバーを示す4個のキャラクター文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

【0202】lengthは、このlengthフィールドの直後からProgramInfo()の最後までのProgramInfo()のバイト数を示す32ビットの符号なし整数である。CPI()のCPI_typeがTU_map typeを示す場合、このlengthフィールドはゼロにセットされても良い。CPI()のCPI_typeがEP_map typeを示す場合、number_of_programsは1以上の値でなければならない。

【0203】number_of_program_sequencesの8ビットの符号なし整数は、Clipの中でのprogram_sequenceの数を示す。この値は、このフィールドに続くfor-loopのループ回数を示す。Clipの中でprogram_sequenceが変化しない場合、number_of_program_sequencesは1をセットされなければならない。RSPN_program_sequence_startの32ビットフィールドは、AVストリームファイル上でプログラムシーケンスが開始する場所の相対アドレスである。

【0204】RSPN_program_sequence_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAVストリームファイルの中での絶対アドレスは、

SPN_xxx = RSPN_xxx - offset_SPN

により算出される。シンタクスのfor-loopの中でRSPN_program_sequence_start値は、昇順に現れなければならない。

【0205】PCR_PIDの16ビットフィールドは、そのprogram_sequenceに有効なPCRフィールドを含むトランスポートパケットのPIDを示す。number_of_videosの8ビットフィールドは、video_stream_PIDとVideoCodingInfo()を含むfor-loopのループ回数を示す。number_of_audiosの8ビットフィールドは、audio_stream_PIDとAudioCodingInfo()を含むfor-loopのループ回数を示す。video_stream_PIDの16ビットフィールドは、そのprogram_sequenceに有効なビデオストリームを含むトランスポートパケットのPIDを示す。このフィールドに続くVideoCodingInfo()は、そのvideo_stream_PIDで参照されるビデオストリームの内容を説明しなければならない。

【0206】audio_stream_PIDの16ビットフィールドは、そのprogram_sequenceに有効なオーディオストリームを含むトランスポートパケットのPIDを示す。このフィールドに続くAudioCodingInfo()は、そのaudio_stream_PIDで参照されるビデオストリームの内容を説明しなければならない。

【0207】なお、シンタクスのfor-loopの中でvideo_stream_PIDの値の現れる順番は、そのprogram_sequenceに有効なPMTの中でビデオストリームのPIDが符号化されている順番に等しくなければならない。また、シンタクスのfor-loopの中でaudio_stream_PIDの値の現れる順番は、そのprogram_sequenceに有効なPMTの中でオーディオストリームのPIDが符号化されている順番に等しくなければならない。

【0208】図55は、図54に示したPrograminfoのシンタクス内のVideoCodingInfoのシンタクスを示す図である。図55に示したVideoCodingInfoのシンタクスを説明するに、video_formatの8ビットフィールドは、図56に示すように、ProgramInfo()の中のvideo_stream_PIDに対応するビデオフォーマットを示す。

【0209】frame_rateの8ビットフィールドは、図57に示すように、ProgramInfo()の中のvideo_stream_PIDに対応するビデオのフレームレートを示す。display_aspect_ratioの8ビットフィールドは、図58に示すように、ProgramInfo()の中のvideo_stream_PIDに対応するビデオの表示アスペクト比を示す。

【0210】図59は、図54に示したPrograminfoのシンタクス内のAudioCodingInfoのシンタクスを示す図である。図59に示したAudioCodingInfoのシンタクスを説明するに、audio_codingの8ビットフィールドは、図60に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオの符号化方法を示す。

【0211】audio_component_typeの8ビットフィールドは、図61に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオのコンポーネントタイプを示す。sampling_frequencyの8ビットフィールドは、図62に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオのサンプリング周波数を示す。

【0212】次に、図45に示したzzzzz.clipのシンタ

クス内のCPI (Characteristic Point Information)について説明する。CPIは、AVストリームの中の時間情報とそのファイルの中のアドレスとを関連づけるためにある。CPIには2つのタイプがあり、それらはEP_mapとTU_mapである。図63に示すように、CPI()の中のCPI_typeがEP_map typeの場合、そのCPI()はEP_mapを含む。図64に示すように、CPI()の中のCPI_typeがTU_map typeの場合、そのCPI()はTU_mapを含む。1つのAVストリームは、1つのEP_mapまたは一つのTU_mapを持つ。AVストリームがSESFトランスポートストリームの場合、それに対応するClipはEP_mapを持たなければならない。

【0213】図65は、CPIのシンタクスを示す図である。図65に示したCPIのシンタクスを説明するに、version_numberは、このCPI()のバージョンナンバーを示す4個のキャラクター文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からCPI()の最後までのCPI()のバイト数を示す32ビットの符号なし整数である。CPI_typeは、図66に示すように、1ビットのフラグであり、ClipのCPIのタイプを表す。

【0214】次に、図65に示したCPIのシンタクス内のEP_mapについて説明する。EP_mapには、2つのタイプがあり、それはビデオストリーム用のEP_mapとオーディオストリーム用のEP_mapである。EP_mapの中のEP_map_typeが、EP_mapのタイプを区別する。Clipが1つ以上のビデオストリームを含む場合、ビデオストリーム用のEP_mapが使用されなければならない。Clipがビデオストリームを含まず、1つ以上のオーディオストリームを含む場合、オーディオストリーム用のEP_mapが使用されなければならない。

【0215】ビデオストリーム用のEP_mapについて図67を参照して説明する。ビデオストリーム用のEP_mapは、stream_PID、PTS_EP_start、および、RSPN_EP_startというデータを持つ。stream_PIDは、ビデオストリームを伝送するトランスポートパケットのPIDを示す。PTS_EP_startは、ビデオストリームのシーケンスヘッダから始めるアクセスユニットのPTSを示す。RSPN_EP_startは、AVストリームの中でPTS_EP_startにより参照されるアクセスユニットの第1バイト目を含むソースポケットのアドレスを示す。

【0216】EP_map_for_one_stream_PID()と呼ばれるサブテーブルは、同じPIDを持つトランスポートパケットによって伝送されるビデオストリーム毎に作られる。Clipの中に複数のビデオストリームが存在する場合、EP_mapは複数のEP_map_for_one_stream_PID()を含んでも良い。

【0217】オーディオストリーム用のEP_mapは、stream_PID、PTS_EP_start、およびRSPN_EP_startというデータを持つ。stream_PIDは、オーディオストリームを伝送するトランスポートパケットのPIDを示す。PTS_EP_startは、オーディオストリームのアクセスユニットのPTSを示す。RSPN_EP_startは、AVストリームの中でPTS_EP_startで参照されるアクセスユニットの第1バイト目を含むソースポケットのアドレスを示す。

【0218】EP_map_for_one_stream_PID()と呼ばれるサブテーブルは、同じPIDを持つトランスポートパケットによって伝送されるオーディオストリーム毎に作られる。Clipの中に複数のオーディオストリームが存在する場合、EP_mapは複数のEP_map_for_one_stream_PID()を含んでも良い。

【0219】EP_mapとSTC_Infoの関係を説明するに、1つのEP_map_for_one_stream_PID()は、STCの不連続点に関係なく1つのテーブルに作られる。RSPN_EP_startの値とSTC_Info()において定義されるRSPN_STC_startの値を比較する事により、それぞれのSTC_sequenceに属するEP_mapのデータの境界が分かる（図68を参照）。EP_mapは、同じPIDで伝送される連続したストリームの範囲に対して、1つのEP_map_for_one_stream_PIDを持たねばならない。図69に示したような場合、program#1とprogram#3は、同じビデオPIDを持つが、データ範囲が連続していないので、それぞれのプログラム毎にEP_map_for_one_stream_PIDを持たねばならない。

【0220】図70は、EP_mapのシンタクスを示す図である。図70に示したEP_mapのシンタクスを説明するに、EP_typeは、4ビットのフィールドであり、図71に示すように、EP_mapのエントリーポイントタイプを示す。EP_typeは、このフィールドに続くデータフィールドのセマンティクスを示す。Clipが1つ以上のビデオストリームを含む場合、EP_typeは0('video')にセットされなければならない。または、Clipがビデオストリームを含まず 1つ以上のオーディオストリームを含む場合、EP_typeは1('audio')にセットされなければならない。

【0221】number_of_stream_PIDsの16ビットのフィールドは、EP_map()の中のnumber_of_stream_PIDsを変数にもつfor-loopのループ回数を示す。stream_PID(k)の16ビットのフィールドは、EP_map_for_one_stream_PID(num_EP_entries(k))によって参照されるk番目のエレメンタリストリーム（ビデオまたはオーディオストリーム）を伝送するトランスポートパケットのPIDを示す。EP_typeが0 ('video')に等しい場合、そのエレメンタリストリームはビデオストリームでなければならない。また、EP_typeが1('audio')に等しい場合、そのエレメンタリストリームはオーディオストリームでなければならない。

【0222】num_EP_entries(k)の16ビットのフィールドは、EP_map_for_one_stream_PID(num_EP_entries(k))によって参照されるnum_EP_entries(k)を示す。EP_map_for_one_stream_PID_Start_address(k): この32ビットのフィールドは、EP_map()の中でEP_map_for_one_s

tream_PID(num_EP_entries(k))が始まる相対バイト位置を示す。この値は、EP_map()の第１バイト目からの大きさで示される。

【０２２３】padding_wordは、EP_map()のシンタクスにしたがって挿入されなければならない。XとYは、ゼロまたは任意の正の整数でなければならない。それぞれのパディングワードは、任意の値を取っても良い。

【０２２４】図７２は、EP_map_for_one_stream_PIDのシンタクスを示す図である。図７２に示したEP_map_for_one_stream_PIDのシンタクスを説明するに、PTS_EP_startの３２ビットのフィールドのセマンティクスは、EP_map()において定義されるEP_typeにより異なる。EP_typeが0 ('video')に等しい場合、このフィールドは、ビデオストリームのシーケンスヘッダで始まるアクセスユニットの３３ビット精度のPTSの上位３２ビットを持つ。EP_typeが1 ('audio')に等しい場合、このフィールドは、オーディオストリームのアクセスユニットの３３ビット精度のPTSの上位３２ビットを持つ。

【０２２５】RSPN_EP_startの３２ビットのフィールドのセマンティクスは、EP_map()において定義されるEP_typeにより異なる。EP_typeが0 ('video')に等しい場合、このフィールドは、AVストリームの中でPTS_EP_startにより参照されるアクセスユニットのシーケンスヘッダの第１バイト目を含むソースポケットの相対アドレスを示す。または、EP_typeが1 ('audio')に等しい場合、このフィールドは、AVストリームの中でPTS_EP_startにより参照されるアクセスユニットのオーディオフレームの第一バイト目を含むソースポケットの相対アドレスを示す。

【０２２６】RSPN_EP_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAVストリームファイルの中での絶対アドレスは、

SPN_xxx = RSPN_xxx - offset_SPN

により算出される。シンタクスのfor-loopの中でRSPN_EP_startの値は、昇順に現れなければならない。

【０２２７】次に、TU_mapについて、図７３を参照して説明する。TU_mapは、ソースパケットのアライバルタイムクロック（到着時刻ベースの時計）に基づいて、１つの時間軸を作る。その時間軸は、TU_map_time_axisと呼ばれる。TU_map_time_axisの原点は、TU_map()の中のoffset_timeによって示される。TU_map_time_axisは、offset_timeから一定の単位に分割される。その単位を、time_unitと称する。

【０２２８】AVストリームの中の各々のtime_unitの中で、最初の完全な形のソースパケットのAVストリームファイル上のアドレスが、TU_mapにストアされる。これらのアドレスを、RSPN_time_unit_startと称する。TU_map_time_axis上において、k (k>=0)番目のtime_unitが始まる時刻は、TU_start_time(k)と呼ばれる。この値は次式に基づいて算出される。

TU_start_time(k) = offset_time + k*time_unit_size

TU_start_time(k)は、45kHzの精度を持つ。

【０２２９】図７５は、TU_mapのシンタクスを示す図である。図７５に示したTU_mapのシンタクスを説明するに、offset_timeの３２bit長のフィールドは、TU_map_time_axisに対するオフセットタイムを与える。この値は、Clipの中の最初のtime_unitに対するオフセット時刻を示す。offset_timeは、２７MHz精度のアライバルタイムクロックから導き出される４５kHzクロックを単位とする大きさである。AVストリームが新しいClipとして記録される場合、offset_timeはゼロにセットされなければならない。

【０２３０】time_unit_sizeの３２ビットフィールドは、time_unitの大きさを与えるものであり、それは２７MHz精度のアライバルタイムクロックから導き出される45kHzクロックを単位とする大きさである。time_unit_sizeは、１秒以下（time_unit_size<=45000）にすることが良い。number_of_time_unit_entriesの３２ビットフィールドは、TU_map()の中にストアされているtime_unitのエントリー数を示す。

【０２３１】RSPN_time_unit_startの３２ビットフィールドは、AVストリームの中でそれぞれのtime_unitが開始する場所の相対アドレスを示す。RSPN_time_unit_startは、ソースパケット番号を単位とする大きさであり、AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAV streamファイルの中での絶対アドレスは、

SPN_xxx = RSPN_xxx - offset_SPN

により算出される。シンタクスのfor-loopの中でRSPN_time_unit_startの値は、昇順に現れなければならない。(k+1)番目のtime_unitの中にソースパケットが何もない場合、(k+1)番目のRSPN_time_unit_startは、k番目のRSPN_time_unit_startと等しくなければならない。

【０２３２】図４５に示したzzzzz.clipのシンタクス内のClipMarkについて説明する。ClipMarkは、クリップについてのマーク情報であり、ClipMarkの中にストアされる。このマークは、記録器（記録再生装置１）によってセットされるものであり、ユーザによってセットされるものではない。

【０２３３】図７５は、ClipMarkのシンタクスを示す図である。図７５に示したClipMarkのシンタクスを説明するに、version_numberは、このClipMark()のバージョンナンバーを示す４個のキャラクター文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

【０２３４】lengthは、このlengthフィールドの直後からClipMark()の最後までのClipMark()のバイト数を示す

３２ビットの符号なし整数である。number_of_Clip_marksは、 ClipMarkの中にストアされているマークの個数を示す１６ビットの符号なし整数。number_of_Clip_marks は、０であってもよい。mark_typeは、マークのタイプを示す８ビットのフィールドであり、図７６に示すテーブルに従って符号化される。

【０２３５】mark_time_stampは、３２ビットフィールドであり、マークが指定されたポイントを示すタイムスタンプをストアする。mark_time_stampのセマンティクスは、図７７に示すように、PlayList()の中のCPI_typeにより異なる。

【０２３６】STC_sequence_idは、CPI()の中のCPI_typeがEP_map typeを示す場合、この８ビットのフィールドは、マークが置かれているところのSTC連続区間のSTC_sequence_idを示す。CPI()の中のCPI_typeがTU_map typeを示す場合、この８ビットのフィールドは何も意味を持たず、ゼロにセットされる。character_setの８ビットのフィールドは、mark_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図１９に示される値に対応する。

【０２３７】name_lengthの８ビットフィールドは、Mark_nameフィールドの中に示されるマーク名のバイト長を示す。mark_nameのフィールドは、マークの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクター文字であり、それはマークの名称を示す。mark_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どんな値が入っていても良い。

【０２３８】ref_thumbnail_indexのフィールドは、マークに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのマークにはサムネイル画像が付加されており、そのサムネイル画像は、mark.thmbファイルの中にストアされている。その画像は、mark.thmbファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、0xFFFF である場合、そのマークにはサムネイル画像が付加されていない。

【０２３９】MakersPrivateDataについては、図２２を参照して既に説明したので、その説明は省略する。

【０２４０】次に、サムネイルインフォメーション（Thumbnail Information）について説明する。サムネイル画像は、menu.thmbファイルまたはmark.thmbファイルにストアされる。これらのファイルは同じシンタクス構造であり、ただ１つのThumbnail()を持つ。menu.thmbファイルは、メニューサムネイル画像，すなわちVolumeを代表する画像、および、それぞれのPlayListを代表する画像をストアする。すべてのメニューサムネイルは、ただ１つのmenu.thmbファイルにストアされる。

【０２４１】mark.thmbファイルは、マークサムネイル画像，すなわちマーク点を表すピクチャをストアする。すべてのPlayListおよびClipに対するすべてのマークサムネイルは、ただ１つのmark.thmbファイルにストアされる。サムネイルは頻繁に追加、削除されるので、追加操作と部分削除の操作は容易に高速に実行できなければならない。この理由のため、Thumbnail()はブロック構造を有する。画像のデータはいくつかの部分に分割され、各部分は一つのtn_blockに格納される。１つの画像データはは連続したtn_blockに格納される。tn_blockの列には、使用されていないtn_blockが存在してもよい。１つのサムネイル画像のバイト長は可変である。

【０２４２】図７８は、menu.thmbとmark.thmbのシンタクスを示す図であり、図７９は、図７８に示したmenu.thmbとmark.thmbのシンタクス内のThumbnailのシンタクスを示す図である。図７９に示したThumbnailのシンタクスについて説明するに、version_numberは、このThumbnail()のバージョンナンバーを示す４個のキャラクター文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

【０２４３】lengthは、このlengthフィールドの直後からThumbnail()の最後までのMakersPrivateData()のバイト数を示す３２ビットの符号なし整数である。tn_blocks_start_addressは、Thumbnail()の先頭のバイトからの相対バイト数を単位として、最初のtn_blockの先頭バイトアドレスを示す３２ビットの符号なし整数である。相対バイト数はゼロからカウントされる。number_of_thumbnailsは、Thumbnail()の中に含まれているサムネイル画像のエントリー数を与える１６ビットの符号なし整数である。

【０２４４】tn_block_sizeは、1024バイトを単位として、１つのtn_blockの大きさを与える16ビットの符号なし整数である。例えば、tn_block_size＝１ならば、それは１つのtn_blockの大きさが1024バイトであることを示す。number_of_tn_blocksは、このThumbnail()中のtn_blockのエントリ数を表す１１６ビットの符号なし整数である。thumbnail_indexは、このthumbnail_indexフィールドから始まるforループ一回分のサムネイル情報で表されるサムネイル画像のインデクス番号を表す１６ビットの符号なし整数である。thumbnail_index として、0xFFFFという値を使用してはならない。thumbnail_index はUIAppInfoVolume()、UIAppInfoPlayList()、PlayListMark()、およびClipMark()の中のref_thumbnail_indexによって参照される。

【０２４５】thumbnail_picture_formatは、サムネイル画像のピクチャフォーマットを表す8ビットの符号なし整数で、図８０に示すような値をとる。表中のDCFとPNGは" menu.thmb" 内でのみ許される。マークサムネイルは、値"0x00" (MPEG-2 Video I-picture)をとらなければならない。

【０２４６】picture_data_sizeは、サムネイル画像のバイト長をバイト単位で示す３２ビットの符号なし整数

である。start_tn_block_numberは、サムネイル画像のデータが始まるtn_blockのtn_block番号を表す１６ビットの符号なし整数である。サムネイル画像データの先頭は、tb_blockの先頭と一致していなければならない。tn_block番号は、０から始まり、tn_blockのfor-ループ中の変数ｋの値に関係する。

【０２４７】x_picture_lengthは、サムネイル画像のフレーム画枠の水平方向のピクセル数を表す１６ビットの符号なし整数である。y_picture_lengthは、サムネイル画像のフレーム画枠の垂直方向のピクセル数を表す１６ビットの符号なし整数である。tn_blockは、 サムネイル画像がストアされる領域である。Thumbnail()の中のすべてのtn_blockは、同じサイズ（固定長）であり、その大きさはtn_block_sizeによって定義される。

【０２４８】図８１は、サムネイル画像データがどのようにtn_blockに格納されるかを模式的に表した図である。図８１のように、各サムネイル画像データはtn_blockの先頭から始まり、1 tn_blockを超える大きさの場合は、連続する次のtn_blockを使用してストアされる。このようにすることにより、可変長であるピクチャデータが、固定長のデータとして管理することが可能となり、削除といった編集に対して簡便な処理により対応する事ができるようになる。

【０２４９】次に、AVストリームファイルについて説明する。AVストリームファイルは、"M2TS"ディレクトリ（図１４）にストアされる。AVストリームファイルには、２つのタイプがあり、それらは、Clip AVストリームとBridge-Clip AVストリームファイルである。両方のAVストリーム共に、これ以降で定義されるDVR MPEG-2トランスポートストリームファイルの構造でなければならない。

【０２５０】まず、DVR MPEG-2 トランスポートストリームについて説明する。DVR MPEG-2トランスポートストリームの構造は、図８２に示すようになっている。AVストリームファイルは、DVR MPEG2トランスポートストリームの構造を持つ。DVR MPEG2トランスポートストリームは、整数個のAligned unitから構成される。Aligned unitの大きさは、6144 バイト (2048*3 バイト)である。Aligned unitは、ソースパケットの第１バイト目から始まる。ソースパケットは、192バイト長である。一つのソースパケットは、TP_extra_headerとトランスポートパケットから成る。TP_extra_headerは、４バイト長であり、またトランスポートパケットは、188バイト長である。

【０２５１】１つのAligned unitは、３２個のソースパケットから成る。DVR MPEG２トランスポートストリームの中の最後のAligned unitも、また３２個のソースパケットから成る。よって、DVR MPEG2トランスポートストリームは、Aligned unitの境界で終端する。ディスクに記録される入力トランスポートストリームのトランスポートパケットの数が３２の倍数でない時、ヌルパケット（PID=0x1FFFのトランスポートパケット）を持ったソースパケットを最後のAligned unitに使用しなければならない。ファイルシステムは、DVR MPEG２トランスポートストリームに余分な情報を付加してはならない。

【０２５２】図８３に、DVR MPEG-2トランスポートストリームのレコーダモデルを示す。図８３に示したレコーダは、レコーディングプロセスを規定するための概念上のモデルである。DVR MPEG-2トランスポートストリームは、このモデルに従う。

【０２５３】MPEG-2トランスポートストリームの入力タイミングについて説明する。入力MPEG2トランスポートストリームは、フルトランスポートストリームまたはパーシャルトランスポートストリームである。入力されるMPEG2トランスポートストリームは、ISO/IEC13818-1またはISO/IEC13818-9に従っていなければならない。MPEG2トランスポートストリームのi番目のバイトは、T-STD (ISO/IEC13818-1で規定されるTransport stream system target decoder)とソースパケッタイザへ、時刻t(i)に同時に入力される。Rpkは、トランスポートパケットの入力レートの瞬時的な最大値である。

【０２５４】２７MHz PLL５２は、２７MHzクロックの周波数を発生する。２７MHzクロックの周波数は、MPEG-2トランスポートストリームのPCR (Program Clock Reference)の値にロックされる。arrival time clock counter５３は、２７MHzの周波数のパルスをカウントするバイナリーカウンターである。Arrival_time_clock(i)は、時刻t(i)におけるArrival time clock counterのカウント値である。

【０２５５】source packetizer５４は、すべてのトランスポートパケットにTP_extra_headerを付加し、ソースパケットを作る。Arrival_time_stampは、トランスポートパケットの第１バイト目がT-STDとソースパケッタイザの両方へ到着する時刻を表す。Arrival_time_stamp(k)は、次式で示されるようにArrival_time_clock(k)のサンプル値であり、ここで、kはトランスポートパケットの第１バイト目を示す。

arrival_time_stamp(k) = arrival_time_clock(k)% 2^{30}

【０２５６】２つの連続して入力されるトランスポートパケットの時間間隔が、2^{30}/27000000秒（約40秒）以上になる場合、その２つのトランスポートパケットのarrival_time_stampの差分は、2^{30}/27000000秒になるようにセットされるべきである。レコーダは、そのようになる場合に備えてある。

【０２５７】smoothing buffer５５は、入力トランスポートストリームのビットレートをスムージングする。スムージングバッファは、オーバーフロウしてはならない。Rmaxは、スムージングバッファが空でない時のスムージングバッファからのソースパケットの出力ビットレートである。スムージングバッファが空である時、スム

ージングバッファからの出力ビットレートはゼロである。

【０２５８】次に、DVR MPEG-2トランスポートストリームのレコーダモデルのパラメータについて説明する。Rmaxという値は、AVストリームファイルに対応するClipInfo()において定義されるTS_recording_rateによって与えられる。この値は、次式により算出される。

Rmax = TS_recording_rate * 192/188

TS_recording_rateの値は、bytes/secondを単位とする大きさである。

【０２５９】入力トランスポートストリームがSESFトランスポートストリームの場合、Rpkは、AVストリームファイルに対応するClipInfo()において定義されるTS_recording_rateに等しくなければならない。入力トランスポートストリームがSESFトランスポートストリームでない場合、この値はMPEG-2 transport streamのデスクリプター，例えばmaximum_bitrate_descriptorやpartial_transport_stream_descriptorなど、において定義される値を参照しても良い。

【０２６０】smoothing buffer sizeは、入力トランスポートストリームがSESFトランスポートストリームの場合、スムージングバッファの大きさはゼロである。入力トランスポートストリームがSESFトランスポートストリームでない場合、スムージングバッファの大きさはMPEG-2 transport streamのデスクリプター、例えばsmoothing_buffer_descriptor、short_smoothing_buffer_descriptor、partial_transport_stream_descriptorなどにおいて定義される値を参照しても良い。

【０２６１】記録機（レコーダ）および再生機（プレーヤ）は、十分なサイズのバッファを用意しなければならない。デフォールトのバッファサイズは、1536 bytesである。

【０２６２】次に、DVR MPEG-2トランスポートストリームのプレーヤモデルについて説明する。図８４は、DVR MPEG-2トランスポートストリームのプレーヤモデルを示す図である。これは、再生プロセスを規定するための概念上のモデルである。DVR MPEG-2トランスポートストリームは、このモデルに従う。

【０２６３】27MHz X-tal６１は、２７Mhzの周波数を発生する。２７MHz周波数の誤差範囲は、+/-30 ppm (27000000 +/- 810 Hz)でなければならない。arrival time clock counter６２は、２７MHzの周波数のパルスをカウントするバイナリーカウンターである。Arrival_time_clock(i)は、時刻t(i)におけるArrival time clock counterのカウント値である。

【０２６４】smoothing buffer６４において、Rmaxは、スムージングバッファがフルでない時のスムージングバッファへのソースパケットの入力ビットレートである。スムージングバッファがフルである時、スムージングバッファへの入力ビットレートはゼロである。

【０２６５】MPEG-2トランスポートストリームの出力タイミングを説明するに、現在のソースパケットのarrival_time_stampがarrival_time_clock(i)のLSB ３０ビットの値と等しい時、そのソースパケットのトランスポートパケットは、スムージングバッファから引き抜かれる。Rpkは、トランスポートパケットレートの瞬時的な最大値である。スムージングバッファは、アンダーフロウしてはならない。

【０２６６】DVR MPEG-2トランスポートストリームのプレーヤモデルのパラメータについては、上述したDVR MPEG-2トランスポートストリームのレコーダモデルのパラメータと同一である。

【０２６７】図８５は、Source packetのシンタクスを示す図である。transport_packet()は、ISO/IEC 13818-1で規定されるMPEG-2トランスポートパケットである。図８５に示したSource packetのシンタクス内のTP_Extra_headerのシンタクスを図８６に示す。図８６に示したTP_Extra_headerのシンタクスについて説明するに、copy_permission_indicatorは、トランスポートパケットのペイロードのコピー制限を表す整数である。コピー制限は、copy free、no more copy、copy once、またはcopy prohibitedとすることができる。図８７は、copy_permission_indicatorの値と、それらによって指定されるモードの関係を示す。

【０２６８】copy_permission_indicatorは、すべてのトランスポートパケットに付加される。IEEE1394デジタルインタフェースを使用して入力トランスポートストリームを記録する場合、copy_permission_indicatorの値は、IEEE1394 isochronouspacket headerの中のEMI (Encryption Mode Indicator)の値に関連付けても良い。IEEE1394デジタルインタフェースを使用しないで入力トランスポートストリームを記録する場合、copy_permission_indicatorの値は、トランスポートパケットの中に埋め込まれたCCIの値に関連付けても良い。アナログ信号入力をセルフエンコードする場合、copy_permission_indicatorの値は、アナログ信号のCGMS-Aの値に関連付けても良い。

【０２６９】arrival_time_stampは、次式

arrival_time_stamp(k) = arrival_time_clock(k)% 2^{30}

において、arrival_time_stampによって指定される値を持つ整数値である。

【０２７０】Clip AVストリームの定義をするに、Clip AVストリームは、上述したような定義がされるDVR MPEG-2トランスポートストリームの構造を持たねばならない。arrival_time_clock(i)は、Clip AVストリームの中で連続して増加しなければならない。Clip AVストリームの中にシステムタイムベース（STCベース）の不連続点が存在したとしても、そのClip AVストリームのarrival_time_clock(i)は、連続して増加しなければならない。

【０２７１】Clip AVストリームの中の開始と終了の間のarrival_time_clock(i)の差分の最大値は、２６時間でなければならない。この制限は、MPEG2トランスポートストリームの中にシステムタイムベース（STCベース）の不連続点が存在しない場合に、Clip AVストリームの中で同じ値のPTS(Presentation Time Stamp)が決して現れないことを保証する。MPEG2システムズ規格は、PTSのラップアラウンド周期を233/90000秒(約26.5時間).と規定している。

【０２７２】Bridge-Clip AVストリームの定義をするに、Bridge-Clip AVストリームは、上述したような定義がされるDVR MPEG-2トランスポートストリームの構造を持たねばならない。Bridge-Clip AVストリームは、１つのアライバルタイムベースの不連続点を含まなければならない。アライバルタイムベースの不連続点の前後のトランスポートストリームは、後述する符号化の制限に従わなければならず、かつ後述するDVR-STDに従わなければならない。

【０２７３】本実施の形態においては、編集におけるPlayItem間のビデオとオーディオのシームレス接続をサポートする。PlayItem間をシームレス接続にすることは、プレーヤ／レコーダに"データの連続供給"と"シームレスな復号処理"を保証する。"データの連続供給"とは、ファイルシステムが、デコーダにバッファのアンダーフロウを起こさせる事のないように必要なビットレートでデータを供給する事を保証できることである。データのリアルタイム性を保証して、データをディスクから読み出すことができるように、データが十分な大きさの連続したブロック単位でストアされるようにする。

【０２７４】"シームレスな復号処理"とは、プレーヤが、デコーダの再生出力にポーズやギャップを起こさせる事なく、ディスクに記録されたオーディオビデオデータを表示できることである。

【０２７５】シームレス接続されているPlayItemが参照するAVストリームについて説明する。先行するPlayItemと現在のPlayItemの接続が、シームレス表示できるように保証されているかどうかは、現在のPlayItemにおいて定義されているconnection_conditionフィールドから判断することができる。PlayItem間のシームレス接続は、Bridge-Clipを使用する方法と使用しない方法がある。

【０２７６】図８８は、Bridge-Clipを使用する場合の先行するPlayItemと現在のPlayItemの関係を示している。図８８においては、プレーヤが読み出すストリームデータが、影をつけて示されている。図８８に示したTS1は、Clip1（Clip AVストリーム）の影を付けられたストリームデータとBridge-ClipのRSPN_arrival_time_discontinuityより前の影を付けられたストリームデータから成る。

【０２７７】TS1のClip1の影を付けられたストリームデータは、先行するPlayItemのIN_time（図８８においてIN_time1で図示されている）に対応するプレゼンテーションユニットを復号する為に必要なストリームのアドレスから、RSPN_exit_from_previous_Clipで参照されるソースパケットまでのストリームデータである。TS1に含まれるBridge-ClipのRSPN_arrival_time_discontinuityより前の影を付けられたストリームデータは、Bridge-Clipの最初のソースパケットから、RSPN_arrival_time_discontinuityで参照されるソースパケットの直前のソースパケットまでのストリームデータである。

【０２７８】また、図８８におけるTS2は、Clip2（Clip AVストリーム）の影を付けられたストリームデータとBridge-ClipのRSPN_arrival_time_discontinuity以後の影を付けられたストリームデータから成る。TS2に含まれるBridge-ClipのRSPN_arrival_time_discontinuity以後の影を付けられたストリームデータは、RSPN_arrival_time_discontinuityで参照されるソースパケットから、Bridge-Clipの最後のソースパケットまでのストリームデータである。TS2のClip2の影を付けられたストリームデータは、RSPN_enter_to_current_Clipで参照されるソースパケットから、現在のPlayItemのOUT_time（図８８においてOUT_time2で図示されている）に対応するプレゼンテーションユニットを復号する為に必要なストリームのアドレスまでのストリームデータである。

【０２７９】図８９は、Bridge-Clipを使用しない場合の先行するPlayItemと現在のPlayItemの関係を示している。この場合、プレーヤが読み出すストリームデータは、影をつけて示されている。図８９におけるTS1は、Clip1（Clip AVストリーム)の影を付けられたストリームデータから成る。TS1のClip1の影を付けられたストリームデータは、先行するPlayItemのIN_time（図８９においてIN_time1で図示されている）に対応するプレゼンテーションユニットを復号する為に必要なストリームのアドレスから始まり、Clip1の最後のソースパケットまでのデータである。また、図８９におけるTS2は、Clip2(Clip AVストリーム)の影を付けられたストリームデータから成る。

【０２８０】TS2のClip2の影を付けられたストリームデータは、Clip2の最初のソースパケットから始まり、現在のPlayItemのOUT_time（図８９においてOUT_time2で図示されている）に対応するプレゼンテーションユニットを復号する為に必要なストリームのアドレスまでのストリームデータである。

【０２８１】図８８と図８９において、TS1とT2は、ソースパケットの連続したストリームである。次に、TS1とTS2のストリーム規定と、それらの間の接続条件について考える。まず、シームレス接続のための符号化制限について考える。トランスポートストリームの符号化構造の制限として、まず、TS1とTS2の中に含まれるプログラムの数は、１でなければならない。TS1とTS2の中に含まれるビデオストリームの数は、１でなければならな

い。TS1とTS2の中に含まれるオーディオストリームの数は、２以下でなければならない。TS1とTS2の中に含まれるオーディオストリームの数は、等しくなければならない。TS1および／またはTS2の中に、上記以外のエレメンタリストリームまたはプライベートストリームが含まれていても良い。

【０２８２】ビデオビットストリームの制限について説明する。図９０は、ピクチャの表示順序で示すシームレス接続の例を示す図である。接続点においてビデオストリームをシームレスに表示できるためには、OUT_time1（Clip1のOUT_time）の後とIN_time2（Clip2のIN_time）の前に表示される不必要なピクチャは、接続点付近のClipの部分的なストリームを再エンコードするプロセスにより、除去されなければならない。

【０２８３】図９０に示したような場合において、BridgeSequenceを使用してシームレス接続を実現する例を、図９１に示す。RSPN_arrival_time_discontinuityより前のBridge-Clipのビデオストリームは、図９０のClip1のOUT_time1に対応するピクチャまでの符号化ビデオストリームから成る。そして、そのビデオストリームは先行するClip1のビデオストリームに接続され、１つの連続でMPEG２規格に従ったエレメンタリストリームとなるように再エンコードされている。

【０２８４】同様にして、RSPN_arrival_time_discontinuity以後のBridge-Clipのビデオストリームは、図９０のClip2のIN_time2に対応するピクチャ以後の符号化ビデオストリームから成る。そして、そのビデオストリームは、正しくデコード開始する事ができて、これに続くClip2のビデオストリームに接続され、１つの連続でMPEG2規格に従ったエレメンタリストリームとなるように再エンコードされている。Bridge-Clipを作るためには、一般に、数枚のピクチャは再エンコードしなければならず、それ以外のピクチャはオリジナルのClipからコピーすることができる。

【０２８５】図９０に示した例の場合にBridgeSequenceを使用しないでシームレス接続を実現する例を図９２に示す。Clip1のビデオストリームは、図９０のOUT_time1に対応するピクチャまでの符号化ビデオストリームから成り、それは、１つの連続でMPEG２規格に従ったエレメンタリストリームとなるように再エンコードされている。同様にして、Clip2のビデオストリームは、図９０のClip2のIN_time2に対応するピクチャ以後の符号化ビデオストリームから成り、それは、一つの連続でMPEG２規格に従ったエレメンタリストリームとなるように再エンコードされている。

【０２８６】ビデオストリームの符号化制限について説明するに、まず、TS1とTS2のビデオストリームのフレームレートは、等しくなければならない。TS1のビデオストリームは、sequence_end_codeで終端しなければならない。TS2のビデオストリームは、Sequence Header、GOP Header、そしてI-ピクチャで開始しなければならない。TS2のビデオストリームは、クローズドGOPで開始しなければならない。

【０２８７】ビットストリームの中で定義されるビデオプレゼンテーションユニット（フレームまたはフィールド）は、接続点を挟んで連続でなければならない。接続点において、フレームまたはフィールドのギャップがあってはならない。接続点において、トップ—ボトムのフィールドシーケンスは連続でなければならない。3-2プルダウンを使用するエンコードの場合は、"top_field_first" および "repeat_first_field"フラグを書き換える必要があるかもしれない，またはフィールドギャップの発生を防ぐために局所的に再エンコードするようにしても良い。

【０２８８】オーディオビットストリームの符号化制限について説明するに、TS1とTS2のオーディオのサンプリング周波数は、同じでなければならない。TS1とTS2のオーディオの符号化方法（例．MPEG1レイヤ2，AC-3，SESF LPCM，AAC）は、同じでなければならない。

【０２８９】次に、MPEG-2トランスポートストリームの符号化制限について説明するに、TS1のオーディオストリームの最後のオーディオフレームは、TS1の最後の表示ピクチャの表示終了時に等しい表示時刻を持つオーディオサンプルを含んでいなければならない。TS2のオーディオストリームの最初のオーディオフレームは、TS2の最初の表示ピクチャの表示開始時に等しい表示時刻を持つオーディオサンプルを含んでいなければならない。

【０２９０】接続点において、オーディオプレゼンテーションユニットのシーケンスにギャップがあってはならない。図９３に示すように、２オーディオフレーム区間未満のオーディオプレゼンテーションユニットの長さで定義されるオーバーラップがあっても良い。TS2のエレメンタリストリームを伝送する最初のパケットは、ビデオパケットでなければならない。接続点におけるトランスポートストリームは、後述するDVR-STDに従わなくてはならない。

【０２９１】ClipおよびBridge-Clipの制限について説明するに、TS1とTS2は、それぞれの中にアライバルタイムベースの不連続点を含んではならない。

【０２９２】以下の制限は、Bridge-Clipを使用する場合にのみ適用される。TS1の最後のソースパケットとTS2の最初のソースパケットの接続点においてのみ、Bridge-ClipAVストリームは、ただ１つのアライバルタイムベースの不連続点を持つ。ClipInfo()において定義されるRSPN_arrival_time_discontinuityが、その不連続点のアドレスを示し、それはTS2の最初のソースパケットを参照するアドレスを示さなければならない。

【０２９３】BridgeSequenceInfo()において定義されるRSPN_exit_from_previous_Clipによって参照されるソースパケットは、Clip1の中のどのソースパケットでも良

い。それは、Aligned unitの境界である必要はない。BridgeSequenceInfo()において定義されるRSPN_enter_to_current_Clipによって参照されるソースパケットは、Clip2の中のどのソースパケットでも良い。それは、Aligned unitの境界である必要はない。

【０２９４】PlayItemの制限について説明するに、先行するPlayItemのOUT_time（図８８、図８９において示されるOUT_time1）は、TS1の最後のビデオプレゼンテーションユニットの表示終了時刻を示さなければならない。現在のPlayItemのIN_time（F図８８、図８９において示されるIN_time2）は、TS2の最初のビデオプレゼンテーションユニットの表示開始時刻を示さなければならない。

【０２９５】Bridge-Clipを使用する場合のデータアロケーションの制限について、図９４を参照して説明するに、シームレス接続は、ファイルシステムによってデータの連続供給が保証されるように作られなければならない。これは、Clip1（Clip AVストリームファイル）とClip2（Clip AVストリームファイル）に接続されるBridge-Clip AVストリームを、データアロケーション規定を満たすように配置することによって行われなければならない。

【０２９６】RSPN_exit_from_previous_Clip以前のClip1（Clip AVストリームファイル）のストリーム部分が、ハーフフラグメント以上の連続領域に配置されているように、RSPN_exit_from_previous_Clipが選択されなければならない。Bridge-Clip AVストリームのデータ長は、ハーフフラグメント以上の連続領域に配置されるように、選択されなければならない。RSPN_enter_to_current_Clip以後のClip2（Clip AVストリームファイル）のストリーム部分が、ハーフフラグメント以上の連続領域に配置されているように、RSPN_enter_to_current_Clipが選択されなければならない。

【０２９７】Bridge-Clipを使用しないでシームレス接続する場合のデータアロケーションの制限について、図９５を参照して説明するに、シームレス接続は、ファイルシステムによってデータの連続供給が保証されるように作られなければならない。これは、Clip1（Clip AVストリームファイル）の最後の部分とClip2（Clip AVストリームファイル）の最初の部分を、データアロケーション規定を満たすように配置することによって行われなければならない。

【０２９８】Clip1（Clip AVストリームファイル）の最後のストリーム部分が、ハーフフラグメント以上の連続領域に配置されていなければならない。Clip2（Clip AVストリームファイル）の最初のストリーム部分が、ハーフフラグメント以上の連続領域に配置されていなければならない。

【０２９９】次に、DVR-STDについて説明する。DVR-STDは、DVR MPEG2トランスポートストリームの生成および検証の際におけるデコード処理をモデル化するための概念モデルである。また、DVR-STDは、上述したシームレス接続された2つのPlayItemによって参照されるAVストリームの生成および検証の際におけるデコード処理をモデル化するための概念モデルでもある。

【０３００】DVR-STDモデルを図９６に示す。図９６に示したモデルには、DVR MPEG-2トランスポートストリームプレーヤモデルが構成要素として含まれている。n, TBn,MBn, EBn, TBsys, Bsys, Rxn, Rbxn, Rxsys, Dn, Dsys, OnおよびPn(k)の表記方法は、ISO/IEC13818-1のT-STDに定義されているものと同じである。すなわち、次の通りである。nは、エレメンタリストリームのインデクス番号である。TBnは、エレメンタリストリームnのトランスポートバッファでる。

【０３０１】MBnは、エレメンタリストリームnの多重バッファである。ビデオストリームについてのみ存在する。EBnは、エレメンタリストリームnのエレメンタリストリームバッファである。ビデオストリームについてのみ存在する。TBsysは、復号中のプログラムのシステム情報のための入力バッファである。Bsysは、復号中のプログラムのシステム情報のためのシステムターゲットデコーダ内のメインバッファである。Rxnは、データがTBnから取り除かれる伝送レートである。Rbxnは、PESパケットペイロードがMBnから取り除かれる伝送レートである。ビデオストリームについてのみ存在する。

【０３０２】Rxsysは、データがTBsysから取り除かれる伝送レートである。Dnは、エレメンタリストリームnのデコーダである。Dsysは、復号中のプログラムのシステム情報に関するデコーダである。Onは、ビデオストリームnのre-ordering bufferである。Pn(k)は、エレメンタリストリームnのk番目のプレゼンテーションユニットである。

【０３０３】DVR-STDのデコーディングプロセスについて説明する。単一のDVR MPEG-2トランスポートストリームを再生している間は、トランスポートパケットをTB1, TBnまたはTBsysのバッファへ入力するタイミングは、ソースパケットのarrival_time_stampにより決定される。TB1, MB1, EB1, TBn, Bn, BsysおよびTBsysのバッファリング動作の規定は、ISO/IEC 13818-1に規定されているT-STDと同じである。復号動作と表示動作の規定もまた、ISO/IEC 13818-1に規定されているT-STDと同じである。

【０３０４】シームレス接続されたPlayItemを再生している間のデコーディングプロセスについて説明する。ここでは、シームレス接続されたPlayItemによって参照される２つのAVストリームの再生について説明をすることにし、以後の説明では、上述した（例えば、図８８に示した）TS1とTS2の再生について説明する。TS1は、先行するストリームであり、TS2は、現在のストリームである。

【0305】図97は、あるAVストリーム（TS1）からそれにシームレスに接続された次のAVストリーム（TS2）へと移る時のトランスポートパケットの入力，復号，表示のタイミングチャートを示す。所定のAVストリーム（TS1）からそれにシームレスに接続された次のAVストリーム（TS2）へと移る間には、TS2のアライバルタイムベースの時間軸（図97においてATC2で示される）は、TS1のアライバルタイムベースの時間軸（図97においてATC1で示される）と同じでない。

【0306】また、TS2のシステムタイムベースの時間軸（図97においてSTC2で示される）は、TS1のシステムタイムベースの時間軸（図97においてSTC1で示される）と同じでない。ビデオの表示は、シームレスに連続していることが要求される。オーディオのプレゼンテーションユニットの表示時間にはオーバーラップがあっても良い。

【0307】DVR-STD への入力タイミングについて説明する。時刻T_1までの時間、すなわち、TS1の最後のビデオパケットがDVR-STDのTB1に入力終了するまでは、DVR-STDのTB1、TBn またはTBsysのバッファへの入力タイミングは、TS1のソースパケットのarrival_time_stampによって決定される。

【0308】TS1の残りのパケットは、TS_recording_rate(TS1)のビットレートでDVR-STDのTBnまたはTBsysのバッファへ入力されなければならない。ここで、TS_recording_rate(TS1)は、Clip1に対応するClipInfo()において定義されるTS_recording_rateの値である。TS1の最後のバイトがバッファへ入力する時刻は、時刻T_2である。従って、時刻T_1からT_2までの区間では、ソースパケットのarrival_time_stampは無視される。

【0309】N1をTS1の最後のビデオパケットに続くTS1のトランスポートパケットのバイト数とすると、時刻T_1乃至T_2までの時間DT1は、N1バイトがTS_recording_rate(TS1)のビットレートで入力終了するために必要な時間であり、次式により算出される。

$\Delta T1 = T_2 - T_1 = N1 / TS\_recording\_rate\ (TS1)$

時刻T_1乃至T_2までの間は、RXnとRXsysの値は共に、TS_recording_rate(TS1)の値に変化する。このルール以外のバッファリング動作は、T-STDと同じである。

【0310】T_2の時刻において、arrival time clock counterは、TS2の最初のソースパケットのarrival_time_stampの値にリセットされる。DVR-STDのTB1，TBn またはTBsysのバッファへの入力タイミングは、TS2のソースパケットのarrival_time_stampによって決定される。RXnとRXsysは共に、T-STDにおいて定義されている値に変化する。

【0311】付加的なオーディオバッファリングおよびシステムデータバッファリングについて説明するに、オーディオデコーダとシステムデコーダは、時刻T_1からT_2までの区間の入力データを処理することができるように、T-STDで定義されるバッファ量に加えて付加的なバッファ量（約1秒分のデータ量）が必要である。

【0312】ビデオのプレゼンテーションタイミングについて説明するに、ビデオプレゼンテーションユニットの表示は、接続点を通して、ギャップなしに連続でなければならない。ここで、STC1は、TS1のシステムタイムベースの時間軸（図97ではSTC1と図示されている）とし、STC2は、TS2のシステムタイムベースの時間軸（図97ではSTC2と図示されている。正確には、STC2は、TS2の最初のPCRがT-STDに入力した時刻から開始する。）とする。

【0313】STC1とSTC2の間のオフセットは、次のように決定される。$PTS^1{}_{end}$は、TS1の最後のビデオプレゼンテーションユニットに対応するSTC1上のPTSであり、$PTS^2{}_{start}$は、TS2の最初のビデオプレゼンテーションユニットに対応するSTC2上のPTSであり、T_{pp}は、TS1の最後のビデオプレゼンテーションユニットの表示期間とすると、2つのシステムタイムベースの間のオフセットSTC_deltaは、次式により算出される。

$STC\_delta = PTS^1{}_{end} + T_{pp} - PTS^2{}_{start}$

【0314】オーディオのプレゼンテーションのタイミングについて説明するに、接続点において、オーディオプレゼンテーションユニットの表示タイミングのオーバーラップがあっても良く、それは0乃至2オーディオフレーム未満である（図97に図示されている"audio overlap"を参照）。どちらのオーディオサンプルを選択するかということと、オーディオプレゼンテーションユニットの表示を接続点の後の補正されたタイムベースに再同期することは、プレーヤ側により設定されることである。

【0315】DVR-STDのシステムタイムクロックについて説明するに、時刻T_5において、TS1の最後のオーディオプレゼンテーションユニットが表示される。システムタイムクロックは、時刻T_2からT_5の間にオーバーラップしていても良い。この区間では、DVR-STDは、システムタイムクロックを古いタイムベースの値（STC1）と新しいタイムベースの値（STC2）の間で切り替える。STC2の値は、次式により算出される。

$STC2 = STC1 - STC\_delta$

【0316】バッファリングの連続性について説明する。$STC1^1{}_{video_end}$は、TS1の最後のビデオパケットの最後のバイトがDVR-STDのTB1へ到着する時のシステムタイムベースSTC1上のSTCの値である。$STC2^2{}_{video_start}$は、TS2の最初のビデオパケットの最初のバイトがDVR-STDのTB1へ到着する時のシステムタイムベースSTC2上のSTCの値である。$STC2^1{}_{video_end}$は、$STC1^1{}_{video\_end}$ の値をシステムタイムベースSTC2上の値に換算した値である。$STC2^1{}_{video_end}$は、次式により算出される。

$STC2^1{}_{video\_end} = STC1^1{}_{video\_end} - STC\_delta$

【0317】DVR-STDに従うために、次の2つの条件を

満たす事が要求される。まず、TS2の最初のビデオパケットのTB1への到着タイミングは、次に示す不等式を満たさなければならない。そして、次に示す不等式を満たさなければならない。

$STC2^2_{video\_start} > STC2^1_{video\_end} + \Delta T1$

この不等式が満たされるように、Clip1および、または、Clip2の部分的なストリームを再エンコードおよび、または、再多重化する必要がある場合は、その必要に応じて行われる。

【0318】次に、STC1とSTC2を同じ時間軸上に換算したシステムタイムベースの時間軸上において、TS1からのビデオパケットの入力とそれに続くTS2からのビデオパケットの入力は、ビデオバッファをオーバーフローおよびアンダーフローさせてはならない。

【0319】このようなシンタクス、データ構造、規則に基づく事により、記録媒体に記録されているデータの内容、再生情報などを適切に管理することができ、もって、ユーザが再生時に適切に記録媒体に記録されているデータの内容を確認したり、所望のデータを簡便に再生できるようにすることができる。

【0320】次に、図46で示したClipInfoのシンタクスの中にあるtime_controlled_flagを1にセットする場合のAVストリームファイルの記録について、詳細な内容を説明する。time_controlled_flagを1にセットする場合、AVストリームの時間経過とAVストリームのデータバイト量が、次の関係にあることを示す。すなわち、AVストリームの時間経過とAVストリームのデータバイト量との関係が、所定の誤差の範囲内で比例する、ことを保証する。

$$TS\_average\_rate * 192/188 * (t - \alpha) <= AV\_file\_size(t) <= TS\_average\_rate * 192/188 * (t + \alpha) \quad \ldots式(1)$$

【0321】上記の式は、図46のClipInfoのtime_controlled_flagの説明の中で示した式とは、すこし形式が違うが本質的には同じである。

【0322】ここで、TS_average_rateは、AVストリームファイル（DVRトランスポートストリームファイル）の平均ビットレートをbytes/second の単位で表したものであり、ClipInfoの中の同名のフィールドにより示される。また、tは、AVストリームファイルの最初のソースパケットからのアライバルライムベースの経過時刻を秒単位で示す。AV_file_size(t)は、 時刻tにおけるAVストリームファイルのサイズをバイト単位で表したものである。αは、所定の一定値であり、例えば、300秒である。

【0323】TS_average_rateは、記録器のアプリケーションによって所定に値に決める。例えば、長時間録画モード（LPモード），標準録画モード（SPモード）、高画質録画モード（HQモード）といった記録モードに応じて、それぞれのモード用のTS_average_rate値を決める。

【0324】式(1)を満たすように、AVストリームファイルが記録されている場合、そのストリームのある時間分だけ部分的にストリームを消去すると、消去した時間分だけ前記ストリームのTS_average_rateで示されるビットレートで記録可能な空き領域をディスク上に作れることを保証できる。例えば、SPモードのAVストリームファイルのある時間分だけ部分的にストリームを消去すると、消去した時間分だけ、同じSPモードで記録可能な空き領域をディスク上に作ることができる。

【0325】図98は、AVストリームの時間経過とAVストリームのデータバイト量との関係が、所定の誤差の範囲内で比例するように、可変ビットレートを制御する場合の、図1の記録再生装置1のAVエンコーダ15の動作を説明するブロック図である。図98と図1で、同じ番号がつけられているブロックは同一のものである。

【0326】まず、ユーザインタフェース24を通して、ユーザーからLP，SPモードなどの記録モードが制御部23に入力される。制御部23は記録モードに応じて、記録するAVストリーム（DVRトランスポートストリーム）の多重化ビットレート、およびビデオ符号化の平均ビットレートを設定する（図99のフローチャートのステップS20参照）。

【0327】制御部23は、time_controlled_flagを1にセットし、多重化ストリームの平均ビットレートをTS_average_rateとし、また多重化ビットレートをTS_recording_rateとする。制御部23は、time_controlled_flag,TS_recording_rateとTS_average_rateをClipInfoに設定したClip Informationファイルのデータベースを出力する。Clip Informationファイルは、図1で説明したようにECC符号化部20の処理を通して、記録媒体に記録される。

【0328】アナログのビデオ入力をエンコードする場合は、端子11からビデオが入力される。または、ディジタル放送入力のビデオをトランスコードする場合は、AVデコーダ27からのビデオが入力される。入力ビデオは、ビデオエンコーダ151へ入力される。制御部23は、所定時間あたりのビデオに対する割り当て符号化ビット量を計算して、それをビデオエンコーダに指定する。ビデオエンコーダ115は、所定時間あたりのビデオをエンコードして、実際に発生した符号化ビット量を制御部23へ入力する。例えば、所定時間の大きさは、ビデオのGOPであり、0.5秒である。制御部23は、エンコーダから入力される実際に発生した符号化ビット量のエンコード開始後の累計値に基づいて、AVストリームの時間経過とAVストリームのデータバイト量

との関係が、所定の誤差の範囲内で比例するように、ビデオ符号化の可変ビットレートの制御をして、次の所定時間あたりのビデオに対する割り当て符号化ビット量を計算する。また、この時に、制御部２３が、エンコーダからビデオの符号化難易度（動きベクトル予測の予測残差の大きさ、ＤＣＴ係数の量子化スケールの大きさ、など）を供給されることができれば、さらに高画質な可変ビットレートを実現できる。すなわち、ビデオの符号化難易度が高いほど、所定時間あたりのビデオに対する割り当て符号化ビット量を大きくするように制御する。

【０３２９】ビデオエンコーダ１１５は、ビデオストリームをマルチプレクサ１６へ入力する。マルチプレクサ１６へはまた、オーディオストリームとAV同期等のシステム情報(S)が入力される。また、オーディオ入力のエンコード処理の流れ、および、AV同期等のシステム情報(S)については、図１の説明と同じである。

【０３３０】マルチプレクサ１６は、ビデオおよびオーディオストリームを、所定の多重化ビットレートのトランスポートストリームに多重化する。この時、ビデオとオーディオのパケット化は、ＭＰＥＧ２トランスポートストリームのシステムターゲットデコーダ（Ｔ－ＳＴＤ）を破綻させないように制御しなければならない。Ｔ－ＳＴＤの制限によって、ビデオのアクセスユニット（符号化されたI，P，Bのピクチャ）およびオーディオのアクセスユニット（オーディオフレーム）をパケット化することができない場合、マルチプレクサ１６は、ヌルパケット（パケットＩＤが、0x1FFFであるパケット）を発生しないように多重化する。この多重化制御により、連続するトランスポートパケットの時間間隔は不規則になり、パケットは間欠的に発生する。

【０３３１】マルチプレクサ１６から出力されるトランスポートパケットは、ソースパケッタイザ１９へ入力される。ソースパケッタイザ１９は、各トランスポートパケットにアライバルタイムスタンプを付加して、ソースパケット化する。そして、ソースパケット列を前詰して、AVストリームファイルを生成する。AVストリームファイルは、図１で説明したようにECC符号化部２０の処理を通して、記録媒体に記録される。

【０３３２】図９９は、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が、所定の誤差の範囲内で比例することを保証する符号化モード（time_controlled_flag=1）において、ビデオを可変ビットレート符号化して、AVストリームを記録する動作を説明するフローチャートである。

【０３３３】ステップＳ２０で、制御部２３は、トランスポートストリームの多重化ビットレートTS_recording_rateおよびビデオ符号化の平均ビットレートを設定する。

【０３３４】ビデオ符号化の平均ビットレートは、TS_average_rateから、オーディオ符号化の一定のビットレートと多重化のオーバヘッドのビットレートを差し引いた値とする。ここで、TS_average_rateは、記録器のアプリケーション（LP，SPモードなど）によって所定に値に決められる。

【０３３５】TS_recording_rateは、ビデオの可変ビットレート符号化の最大ビットレートに、オーディオ符号化の一定のビットレートと多重化のオーバヘッドのビットレートを加えた値よりも大きい値である。

【０３３６】ステップＳ２１で、制御部２３は、ビデオストリームを、あらかじめ設定した所定の時間区間毎に所定の平均ビットレートが保証される様に、可変ビットレートでエンコードするようにビデオエンコーダ１５１を制御する。

【０３３７】ステップＳ２２で、制御部２３は、トランスポートパケット化するエレメンタリストリームがない場合にヌルパケットを発生しないようにマルチプレクサ１６を制御する。この多重化制御により、連続する2個のトランスポートパケットの時間間隔は不規則になり、パケットは間欠的に発生する。

【０３３８】ステップＳ２３で、制御部２３は、各トランスポートパケットにアライバルタイムスタンプを付加して、ソースパケット化するように、ソースパケッタイザ１９を制御し、そして、ソースパケット列を前詰して、AVストリームファイルとして記録するように制御する。

【０３３９】次に、ビデオの可変ビットレート符号化をする場合のＭＰＥＧのＶＢＶ（Ｖｉｄｅｏ Ｂｕｆｆｅｒｉｎｇ Ｖｅｒｉｆｉｅｒ）の制御方法について説明する。ＶＢＶは、MPEGが規定する理論的なデコーダモデルである（図１００を参照）。ＭＰＥＧビデオエンコーダは、ＶＢＶを正しく動作させるようにビデオストリームをエンコードしなければならない。これにより、エンコード方法を制限する（主に量子化制御およびピクチャのビット量の制限）。VBVの持つバッファをVBVバッファと呼ぶ。これは現実のデコーダに理論上、最低必要なバッファサイズである。ＭＰＥＧ２メインプロファイルメインレベルの場合、VBVバッファサイズは、1.75 Mbitsである。

【０３４０】可変ビットレート時のＭＰＥＧのＶＢＶは、一般に、図１０１で示す方法が広く知られている。すなわち、図１０１は、VBVバッファに空きがあるときは、バッファへの入力ビットレートがVBR(Variable Bit-Rate、可変ビットレート)の最大ビットレートであり、VBVバッファのビット占有量がフルの場合は、バッファへの入力ビットレートがゼロになる場合のVBV制御を説明する図である。図１０１において、右上がりの線の傾きは、VBRの最大ビットレートを示し、VBVバッファに空きがあるときは、VBRの最大ビットレートでバッファ占有量が増える。また、VBVバッファのビット占有量がフルの場合は、バッファへの入力ビットレートがゼロとな

り、バッファ占有量は変わらない。横軸は時間軸であり、T1は一つのデコード時刻を示し、時刻T_1において図示するT_1の時刻のピクチャが瞬時にデコードされて、バッファ占有量が減少する。以後、所定の時間間隔で同様にして、ピクチャがデコードされて、バッファ占有量が減少する。この図１０１で示す方法では、ビデオエンコーダがビデオストリーム中にスタッフィングバイトを発生することはない。

【０３４１】これに対して、本発明では、ＶＢＶを図１０２に示すように制御する。すなわち、所定の時間（例えば、ＧＯＰ）毎にビットレートを変更する可変ビットレートにおいて、所定の時間内ではCBR(Constant Bit-Rate、固定ビットレート)のＶＢＶ制御を行う。図１０２は、GOP（例えば、0.5秒のビデオシーケンス）内でCBRの場合のVBV制御を示す。すなわち、VBVバッファへの入力ビットレートが、現在のGOPの符号化ビットレートであり、VBVバッファがオーバーフローしないようにスタッフィングバイトを挿入する場合のVBV制御を説明する図である。

【０３４２】スタッフィングバイトの挿入するかどうかの判断と、挿入する場合のスタッフィングバイトの量の計算は、次の手順で行う。以下の説明において、

VBV_BUFFER_SIZE = 1.75*1024*1024 bit

gop_bit_rate: GOP毎のビットレート〔bit/second〕

とする。

【０３４３】(1) 現在、符号化するピクチャの最低ビット量の計算。図１０２の時刻d1のピクチャを例として説明する。まず、時刻d1のピクチャをＶＢＶがデコードする直前のＶＢＶバッファのビット占有量vbv_bを得る。次に、ビット占有量vbv_bに、時刻d1からその次のピクチャのデコード時刻d2までの間（tau）にビットレートgop_bit_rateで入力されるビット量を加えた値tmpを計算する。現在、符号化するピクチャの最低ビット量min_picture_bitは、tmp と VBV_BUFFER_SIZEから次のように計算できる。

tmp = vbv_b + gop_bit_rate*tau

min_picture_bit = tmp - VBV_BUFFER_SIZE

【０３４４】(2) pictureの符号化後に、byte stuffingが必要かのチェック。現在のピクチャの実際の符号化ビットgen_picture_bitが、min_picture_bitより小さい場合は、次に示す計算式で示す大きさのスタッフィングバイト発生する。現在符号化したpictureの後にnum_stuffing_byteの数のstuffing bytesをビデオエンコーダが符号化する。一つのスタッフィングバイトは、8ビットの"0000 0000"の符号である。

if (gen_picture_bit < min_picture_bit)

num_stuffing_byte=(min_picture_bit-gen_picture_bit+4)/8

【０３４５】この図１０２で示す方法では、ビデオエンコーダが所定時間のビデオに割り当てられたビット量を使うように制御することを目的として、VBVバッファへの入力ビットレートが現在のGOPの符号化ビットレートであり、VBVバッファがオーバーフローしないようにビデオエンコーダがスタッフィングバイトを発生する。

【０３４６】図１０２に示すＶＢＶ制御は、本発明のコンセプトである、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が図１０３に示すように、所定の誤差範囲内で比例することを保証するために、有効である。図１０１に示すＶＢＶ制御を使うと、入力ビデオの中に長い時間の静止画像があると、図１０３の関係を保証できなくなる。すなわち、静止画像は情報量が比較的小さいため、その情報量よりも符号化の割り当てビット量を大きくしても、実際に符号化して発生するビット量はある比較的小さな値に飽和してしまう。したがって、この場合、ＡＶストリームの時間経過とＡＶストリームのデータバイト量の関係が図１０４に示すように、比例しない。このような場合でも、図１０２に示すＶＢＶ制御を使えば、ビデオエンコーダが所定時間のビデオに割り当てられたビット量を使うように制御することを目的として、VBVバッファへの入力ビットレートが現在のGOPの符号化ビットレートであり、VBVバッファがオーバーフローしないようにビデオエンコーダがスタッフィングバイトを発生するので、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が図１０３に示すように、所定の誤差範囲内でほぼ比例することを保証できる。

【０３４７】図１０４の場合、静止画像部分の時間部分のＡＶストリームを消去しても、その部分の占めるデータバイト量は、平均ビットレートに消去時間を掛けたデータサイズよりも小さいため、消去した時間分だけ前記ストリームのTS_average_rateで示されるビットレートで記録可能な空き領域をディスク上に作れることができない。一方、図１０３の場合、ＡＶストリームのある時間分だけ部分的にストリームを消去すると、消去した時間分だけ前記ストリームのTS_average_rateで示されるビットレートで記録可能な空き領域をディスク上に作れることができる。

【０３４８】図１０５は、上述の図９９のステップS21の処理における、ビデオの可変ビットレート制御の処理の詳細を説明するフローチャートである。

【０３４９】ステップＳ２００で、VBRの余裕量sv_nowに初期値SV1をセットする。本発明の可変ビットレート制御は、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が所定の誤差範囲内で比例することを保証するために、VBRの余裕量sv_nowが、ゼロから最大値SVMAXになるように制御を行う。

【０３５０】例えば、上記の式（１）において、α＝３００秒の場合、SV1, SVMAXは次の値である。ここで、ビデオの平均符号化ビットレートは、図９９のステップＳ２０で決定された値である（図１０７を参照）。

SV1　= (ビデオの平均符号化ビットレート) * 300
SVMAX = SV1 * 2
【0351】ステップS201で、現GOPの符号化の割り当てビットb_allocの計算する。
【0352】ステップS202で、以下の不等式が成り立つかを調べる。このステップSは、VBRの余裕量がマイナスにならないかどうかチェックである。
sv_now + b_av - b_alloc >= 0
【0353】ここで、b_avは、ビデオの平均符号化ビットレートから計算される、GOPあたりの符号化の割り当てビット量の平均値である。GOPの時間長を、0.5秒とするとb_avは次の値である。
b_av = (ビデオの平均ビットレート) * 0.5
【0354】ステップS202でYesの場合は、ステップS203へ進む。ステップS202でNoの場合は、ステップS204へ進み、b_allocをb_avとし、ステップS205へ進む。
【0355】ステップS203では、以下の不等式が成り立つかを調べる。このステップSは、VBRの余裕量が最大値SVMAXを超えないかどうかチェックである。
sv_now + b_av - b_alloc <= SVMAX
【0356】ステップS203でYesの場合は、ステップS205へ進む。ステップS203でNoの場合は、ステップS204へ進み、b_allocをb_avとし、ステップS205へ進む。
【0357】ステップS205で、現在のGOPのエンコードする。そして、現在のGOPを割り当てビット量b_allocでエンコードし、その時のVBV制御は、VBVバッファへの入力ビットレートを現在のGOPの符号化ビットレートとし、VBVバッファがオーバーフローしないようにスタッフィングバイトを挿入するように制御する。この処理の詳細については、図106で説明する。
【0358】ステップS206で、VBRの余裕量sv_nowを次式のように更新する。ここで、b_genは、ステップS205で、現在のGOPのエンコードした結果、得られた現GOPの符号化ビット量である。
sv_now += b_av - b_gen
【0359】ステップS207で、現GOPが最後のGOPであるか調べる。ステップS207で、Yesの場合は、処理を終了する。ステップS207で、Noの場合は、ステップS201へ戻る。
【0360】図106は、上述の図105のステップS205の処理における、VBV制御の処理の詳細を説明するフローチャートである。
【0361】ステップS300で、次式のように現GOPに割り当てられた符号化ビット量を符号化ビットレートgop_bit_rateに変換する。
gop_bit_rate = b_alloc / (15/ 29.97)
【0362】ステップS301で、現GOPの中で、現在符号化するピクチャの最低ビット量min_picture_bitを次式により計算する。
tmp = vbv_b　+ gop_bit_rate*tau
min_picture_bit = tmp - VBV_BUFFER_SIZE
【0363】ここで、vbv_bは、VBVが、現在符号化するピクチャをデコードする直前のVBVバッファのビット占有量である(図102参照)。
【0364】tauは、現在 符号化するピクチャのデコード時刻と その次のピクチャのデコード時刻の差である(図102参照)。
【0365】VBV_BUFFER_SIZEは、VBVバッファサイズであり、MPEG2　MP@MLの場合、1.75 Mbitである。
【0366】ステップS302で、現在のピクチャのエンコードし、その発生ビット量gen_picture_bitを得る。
【0367】ステップS303で、次の不等式を調べる。
gen_picture_bit < min_picture_bit
【0368】ステップS303でYesの場合は、ステップS304へ進む。ステップS303でNoの場合は、ステップS305へ進む。
【0369】ステップS304で、現在符号化したpictureの後にnum_stuffing_byteの数のスタッフィングバイトをビデオエンコーダが現在符号化し、それらを符号化ピクチャの後ろに付加する(図102参照)。
num_stuffing_byte=(min_picture_bit-gen_picture_bit+4)/8
【0370】ステップS305で、GOPの最後のピクチャかどうか調べる。ステップS305で、Yesの場合は、処理を終了する。ステップS305で、Noの場合は、ステップS301へ戻る。
【0371】以上のようにして、ビデオストリームの可変ビットレート符号化を制御し、AVストリームファイルを生成することにより、AVストリームの時間経過とAVストリームのデータバイト量との関係が、所定の誤差の範囲内で比例することを保証することできる。これにより、そのストリームのある時間分だけ部分的にストリームを消去すると、消去した時間分だけ前記ストリームのTS_average_rateで示されるビットレートで記録可能な空き領域をディスク上に作れることを保証できる。
【0372】次に、比較のため、AVストリームの時間経過とAVストリームのデータバイト量との関係が比例することを保証しない符号化モード(time_controlled_flag=0)におけるAVストリームの記録方法の例を2つ示す。
【0373】一つ目のtime_controlled_flag=0の場合の例は、ディジタル放送のAVストリーム(プログラム)のトランスポートストリームをトランスペアレント記録する場合である。ディジタル放送が統計多重を用いている場合、一般に、その中のAVストリームは可変ビット

レートである。一般に、この場合のＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が比例することは保証されないので、このＡＶストリームをトランスペアレント記録してClipを作成した場合、そのClipのtime_controlled_flagをゼロにセットする。

【０３７４】二つ目のtime_controlled_flag=0の場合の例は、ビデオを可変ビットレート符号化する場合に、ビデオストリームを、あらかじめ設定した所定の時間区間毎に所定の平均ビットレート以下になる様に、可変ビットレートでエンコードする場合である。これは、図１０１で説明したように、ビデオ符号化のVBV制御が、VBVバッファに空きがあるときは、バッファへの入力ビットレートをVariable Bit-Rateの最大ビットレートにし、VBVバッファのビット占有量がフルの場合は、バッファへの入力ビットレートをゼロにする場合である。図１０８と図１０９を用いて、この場合のＡＶストリームの記録方法を説明する。

【０３７５】図１０８は、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が、比例することを保証しない符号化モード（time_controlled_flag=0）において、ビデオを可変ビットレート符号化して、AVストリームを記録する動作を説明するフローチャートを示す。

【０３７６】ステップＳ４００以外は、図９９と同じである。

【０３７７】ステップＳ４００で、ビデオストリームを、あらかじめ設定した所定の時間区間毎に所定の平均ビットレート以下になる様に、可変ビットレートでエンコードするようにビデオエンコーダ１５１を制御する。

【０３７８】図１０９は、上述の図１０８のステップＳ４００の処理における、ビデオの可変ビットレート制御の処理の詳細を説明するフローチャートである。

【０３７９】ステップＳ５００で、VBRの余裕量sv_nowに初期値SV1をセットする。この場合の可変ビットレート制御は、VBRの余裕量sv_nowが、負の値にならないように制御を行う。

【０３８０】ステップＳ５０１で、現GOPの符号化の割り当てビットb_allocの計算する。

【０３８１】ステップＳ５０２で、以下の不等式が成り立つかを調べる。このステップSは、VBRの余裕量がマイナスにならないかどうかチェックである。

sv_now + b_av - b_alloc >= 0

【０３８２】ここで、b_avは、ビデオの平均符号化ビットレートから計算される、GOPあたりの符号化の割り当てビット量の平均値である。ＧＯＰの時間長を、0.5秒とするとb_avは次の値である。

b_av = （ビデオの平均ビットレート）* 0.5

【０３８３】ステップＳ５０２でＹｅｓの場合は、ステップＳ５０４へ進む。ステップＳ５０２でＮｏの場合は、ステップＳ５０４へ進み、b_allocをb_avとし、ステップＳ５０４へ進む。

【０３８４】ステップＳ５０４で、現在のGOPのエンコードする。そして、現在のGOPを割り当てビット量b_allocでエンコードし、その時のVBV制御は、その時のVBV制御は、VBVバッファに空きがあるときは、バッファへの入力ビットレートをVBR(Variable Bit-Rate)の最大ビットレートにし、VBVバッファのビット占有量がフルの場合は、バッファへの入力ビットレートをゼロにする場合のVBV制御とする（図１０１参照）。このステップＳでは、ビデオストリームにスタッフィングバイトを符号化しない。

【０３８５】ステップＳ５０５で、ＶＢＲの余裕量sv_nowを次式のように更新する。ここで、b_genは、ステップＳ５０４で、現在のGOPのエンコードした結果、得られた現GOPの符号化ビット量である。

sv_now += b_av - b_gen

【０３８６】ステップＳ５０６で、現GOPが最後のGOPであるか調べる。ステップＳ５０６で、Ｙｅｓの場合は、処理を終了する。ステップＳ５０６で、Ｎｏの場合は、ステップＳ５０１へ戻る。

【０３８７】上記の図１０８および図１０９の記録方法の場合、前述したようにＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が所定の誤差範囲内で比例することを保証しない。例えば、入力ビデオの中に長い時間の静止画像があると、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が図１０４に示したようになる。すなわち、静止画像は情報量が比較的小さいため、その情報量よりも符号化の割り当てビット量を大きくしても、実際に符号化して発生するビット量はある比較的小さな値に飽和してしまう。したがって、この場合、ＡＶストリームの時間経過とＡＶストリームのデータバイト量の関係が、比例しない。

【０３８８】一方、ビデオエンコーダが所定時間のビデオに割り当てられたビット量を使うように制御することを目的として、VBVバッファへの入力ビットレートが現在のGOPの符号化ビットレートであり、VBVバッファがオーバーフローしないようにビデオエンコーダがスタッフィングバイトを発生するように制御すれば、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が、所定の誤差範囲内でほぼ比例することを保証できる。

【０３８９】また、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が、比例することを保証する符号化モード（time_controlled_flag=1）を簡単に実現する方法として、トランスポートストリームを多重化する時にヌルパケットを挿入して、一定ビットレートのトランスポートストリームを記録することも考えられる。これは、主にテープ記録媒体（Ｄ－ＶＨＳ等）で用いられている符号化方法である。ここで、ヌルパケットは、そのパケットＩＤ（ＰＩＤ）が、0x1FFFにセッ

トされている、情報としては何も意味をもたないトランスポートパケットである。

【０３９０】図９９の方法と比較する参考のために、図１１０に、所定の一定ビットレートのトランスポートストリームを符号化することによって、ＡＶストリームの時間経過とＡＶストリームのデータバイト量との関係が、比例することを保証する符号化モードのフローチャートを示す。

【０３９１】ステップＳ600で、トランスポートストリームの多重化ビットレートおよびビデオ符号化のビットレートを設定する　ステップＳ601で、ビデオストリームを、所定の一定のビットレート、または、そのビットレート以下で、エンコードする。

【０３９２】ステップＳ602で、トランスポートパケット化するエレメンタリストリームがない場合にヌルパケット（情報としては意味をもたないトランスポートパケット）を発生して多重化し、所定の一定の多重化ビットレートのトランスポートストリームを符号化する。

【０３９３】ステップＳ603で、各トランスポートパケットにアライバルタイムスタンプを付加して、ソースパケット化する。ソースパケットを記録媒体に記録する。

【０３９４】上記の記録方法でＡＶストリームをClipとして記録した場合、そのClipのtime_controlled_flagは１にセットされる。しかしながら、この方法は、ヌルパケットを使用するため、ビデオ符号化に効率良く符号ビットを使用していないので、図９９の符号化方法よりもビデオの画質が劣る問題がある（このことについては、例えば特願平11-220727の従来の技術の欄に詳しく述べている）。そのため、本発明では上記の図１１０の記録方法を推奨しない。

【０３９５】次に、ＡＶストリームファイルのある時間分だけ部分的にストリームを消去する方法について説明する。

【０３９６】図１１１は、オリジナルのＡＶストリームファイルと、そのストリームの部分的な再生範囲のストリームを消去する編集を行った後のＡＶストリームファイルの例を示す。編集前に、Virtual PlayListは、オリジナルAVストリーム上のIN_timeとOUT_timeを指しているとする。この時、Virtual PlayListが使用していないストリーム部分を消去する編集（ミニマイズ編集）をした場合、それはオリジナルAVストリームを図１１１に示す編集後のストリームへ変える。オリジナルAVストリームの先頭からＸ点までのデータと、Ｙ点から最後までのデータが消去される。以下の説明では、このＸ点とＹ点を決める方法の例を説明する。

【０３９７】図１１２は、AVストリームの内容を解析することをしないで、IN点の前の不要なデータを消去する方法を説明する図である。PlayListはオリジナルAVストリーム上のIN点を指す。また、そのAVストリームのEP_mapを図示する。IN点が指すピクチャをデコードするためには、アドレスISA2から開始するＩピクチャが必要である。

【０３９８】また、Ｘ点の後で、PAT,PMTおよびPCRパケットが必要である。RSPN_EP_start=ISA1のPTSはpts1であり、RSPN_EP_start=ISA2のPTSはpts2である。pts1とpts2のシステムタイムベースの時間差が100 msec以上ならば、アドレスISA1とISA2の間にはPAT, PMTおよびPCRパケットが存在する（少なくとも、SESF, DVB, ATSC, ISDBの場合はそうである）。

【０３９９】したがって、Ｘ点はアドレスISA1の前に決められる。そして、X点はアラインドユニットの境界でなければならない。記録装置は、AVストリームの内容を解析することをしないで、X点をEP_mapを使用して次のステップＳで決めることができる。

（Ｓ１）システムタイムベース上でIN timeのPTSに最も近く、かつそれよりも過去の表示時刻のPTSの値を持つSPN_EP_startを見つける。

（Ｓ２）ステップＳ1で見つけたSPN_EP_startのPTSの値よりも少なくとも100 msec過去の表示時刻のPTSの値を持つSPN_EP_startを見つける。

（Ｓ３）Ｘ点は、ステップＳ２で見つけたSPN_EP_startよりも前に決められる。そして、Ｘ点はアラインドユニットの境界でなければならない。

【０４００】この方法は、Ｘ点を決めるためにＡＶストリームのデータを読み出し、その内容を解析することを必要としないので、簡単である。しかし、編集後のAVストリームは、そのPlayListの再生には不要なデータを残してしまう場合がある。もし、Ｘ点を決めるためにＡＶストリームのデータを読み出し、その内容を解析するならば、そのPlayListの再生には不要なデータをより効率良く消去できる。

【０４０１】図１１３は、AVストリームの内容を解析することをしないで、OUT点の後ろの不要なデータを消去する方法を説明する図である。PlayListはオリジナルAVストリーム上のOUT点を指す。また、そのAVストリームのEP_mapを図示する。

【０４０２】SPN_EP_start=ISA4から開始するビデオシーケンスは次に示すものであることを前提とする。

I2 B0 B1 P5 …

ここで、I,P,Bはそれぞれ I ピクチャ, P ピクチャそして B ピクチャを表す。数字は表示順序を表す。この処理において、記録装置がAVストリームの内容を解析しない場合、記録装置はOUT_timeのPTSが参照するところのピクチャの情報（ピクチャコーディングタイプ, テンポラルレファレンスなど）がわからない。OUT_timeのPTSはピクチャB0またはB1を参照しているかもしれない（記録装置がAVストリームの内容を解析しない場合、このことはわからない）、この場合、ピクチャB0,B1をデコードするためにはI2が必要である。I2のPTSはOUT timeのPTSよりも大きい（OUT_time < pts4, ここでpts4はI2のPTSで

ある)。I2のPTSはOUT_timeのPTSよりも大きいが、B0,B1のためにI2が必要である。

【0403】したがって、Y点は図に示すアドレスISA5の後ろに決められる。ISA5は、EP_mapの中でISA4の直後にあるSPN_EP_startの値である。Y点はまたアラインドユニットの境界でなければならない。

【0404】記録装置は、AVストリームの内容を解析することをしないで、Y点をEP_mapを使用して次のステップSで決めることができる。

(S1)システムタイムベース上でOUT timeのPTSに最も近く、かつそれよりも未来の表示時刻のPTSの値を持つSPN_EP_startを見つける。

(S2)ステップS1で見つけたSPN_EP_startの直後にあるSPN_EP_start を見つける。

(S3)Y点は、ステップS2で見つけたSPN_EP_startよりも後ろに決められる。そして、Y点はアラインドユニットの境界でなければならない。

【0405】この方法は、Y点を決めるためにAVストリームのデータを読み出し、その内容を解析することを必要としないので、簡単である。しかし、編集後のAVストリームは、そのPlayListの再生には不要なデータを残してしまう場合がある。もし、Y点を決めるためにAVストリームのデータを読み出し、その内容を解析するならば、そのPlayListの再生には不要なデータをより効率良く消去できる。

【0406】次に、EP_mapの作成の動作例を図114のフローチャートを用いて説明する。この処理は図1の記録再生装置の多重化ストリーム解析部18で行われる。

【0407】ステップS11でストリーム解析部18は、記録するAVプログラムのビデオのPIDをセットする。トランスポートストリームの中に複数のビデオが含まれている場合は、それぞれのビデオPIDをセットする。

【0408】ステップS12でストリーム解析部18は、ビデオのトランスポートパケットを受信する。

【0409】ステップS13でストリーム解析部は、トランスポートパケットのペイロード(パケットヘッダーに続くデータ部)がPESパケットの第一バイト目から開始しているかを調べる(PESパケットは、MPEG2で規定されているパケットであり、エレメンタリストリームをパケット化するものである)。これは、トランスポートパケットヘッダにある"payload_unit_start_indicator"の値を調べることによりわかり、この値が1である場合、トランスポートパケットのペイロードがPESパケットの第一バイト目から開始する。ステップS13でNoの場合は、ステップS12へ戻り、Yesの場合は、ステップS14へ進む。

【0410】ステップS14でストリーム解析部は、PESパケットのペイロードが、MPEGビデオのsequence_header_code(32ビット長で"0x000001B3"の符号)の第一バイト目から開始しているかを調べる。ステップS14でNoの場合は、ステップS12へ戻り、Yesの場合は、ステップS15へ進む。

【0411】ステップS15へ進んだ場合、現在のトランスポートパケットをエントリーポイントとする。ステップS16でストリーム解析部は、上記パケットのパケット番号と上記sequence_header_code から開始するIピクチャのPTSとそのエントリーポイントが属するビデオのPIDを取得し、制御部23へ入力する。制御部23はEP_mapを作成する。

【0412】ステップS17で、現在のパケットが最後に入力されるトランスポートパケットであるかどうかを判定する。最後のパケットでない場合、ステップS12へ戻る。最後のパケットである場合、処理を終了する。

【0413】上述した一連の処理は、ハードウェアにより実行させることもできるが、ソフトウェアにより実行させることもできる。一連の処理をソフトウェアにより実行させる場合には、そのソフトウェアを構成するプログラムが専用のハードウェアに組み込まれているコンピュータ、または、各種のプログラムをインストールすることで、各種の機能を実行することが可能な、例えば汎用のパーソナルコンピュータなどに、記録媒体からインストールされる。

【0414】この記録媒体は、図115に示すように、コンピュータとは別に、ユーザにプログラムを提供するために配布される、プログラムが記録されている磁気ディスク221(フロッピディスクを含む)、光ディスク222(CD-ROM(Compact Disk-Read Only Memory),DVD(Digital Versatile Disk)を含む)、光磁気ディスク223(MD(Mini-Disk)を含む)、若しくは半導体メモリ224などよりなるパッケージメディアにより構成されるだけでなく、コンピュータに予め組み込まれた状態でユーザに提供される、プログラムが記憶されているROM202や記憶部208が含まれるハードディスクなどで構成される。

【0415】なお、本明細書において、媒体により提供されるプログラムを記述するステップSは、記載された順序に従って、時系列的に行われる処理は勿論、必ずしも時系列的に処理されなくとも、並列的あるいは個別に実行される処理をも含むものである。

【0416】また、本明細書において、システムとは、複数の装置により構成される装置全体を表すものである。

【0417】

【発明の効果】以上の如く、AVストリームを符号化して記録する時に、そのAVストリームの属性情報として、time_controlled_flag, TS_average_rateを記録する。time_controlled_flagを1にセットする場合、AVストリームの時間経過とAVストリームのデータバイト量との関係が、所定の誤差の範囲内で比例することを保証する。また、TS_average_rateは、AVストリームファ

イル（トランスポートストリーム）の平均ビットレートをbytes/second の単位で表したものである。TS_average_rateは、記録器のアプリケーションによって所定に値に決める。例えば、長時間録画モード（ＬＰモード），標準録画モード（ＳＰモード）、高画質録画モード（ＨＱモード）といった記録モードに応じて、それぞれのモードのTS_average_rateの値を決める。

【０４１８】ＡＶストリームファイルのtime_controlled_flagが1にセットされている場合、そのストリームのある時間分だけ部分的にストリームを消去すると、消去した時間分だけ前記ストリームのTS_average_rateで示されるビットレートで記録可能な空き領域をディスク上に作れることを保証できる。例えば、ＳＰモードのＡＶストリームファイルのある時間分だけ部分的にストリームを消去すると、消去した時間分だけ、同じＳＰモードで記録可能な空き領域をディスク上に作ることができる。

【０４１９】time_controlled_flagを1にセットする場合、次のようにしてＡＶストリームを符号化する。

（１）トランスポートストリームの多重化ビットレートおよびビデオ符号化の平均ビットレートを設定する。

（２）ビデオストリームを、あらかじめ設定した所定の時間区間毎に所定の平均ビットレートが保証される様に、可変ビットレートでエンコードする。ここで、MPEGビデオ符号化のＶＢＶ(Video Buffering Verifier)制御は、ビデオエンコーダが所定時間のビデオに割り当てられたビット量を使うように制御することを目的として、VBVバッファへの入力ビットレートが現在の符号化ビットレートであり、VBVバッファがオーバーフローしないようにビデオエンコーダがスタッフィングバイトを発生するようにする。

（３）トランスポートパケット化するエレメンタリストリームがない場合にヌルパケットを発生しないように多重化の制御をする。

（４）各トランスポートパケットにアライバルタイムスタンプを付加して、ソースパケット化し、そして、ソースパケット列を前詰して、AVストリームファイルとして記録する。

【０４２０】このようにして、ＡＶストリームを符号化して記録することにより、そのストリームのある時間分だけ部分的にストリームを消去すると、消去した時間分だけ前記ストリームのTS_average_rateで示されるビットレートで記録可能な空き領域をディスク上に作れることを保証できる。

【図面の簡単な説明】

【図１】本発明を適用した記録再生装置の一実施の形態の構成を示す図である。

【図２】記録再生装置１により記録媒体に記録されるデータのフォーマットについて説明する図である。

【図３】Real PlayListとVirtual PlayListについて説明する図である。

【図４】Real PlayListの作成について説明する図である。

【図５】Real PlayListの削除について説明する図である。

【図６】アセンブル編集について説明する図である。

【図７】Virtual PlayListにサブパスを設ける場合について説明する図である。

【図８】PlayListの再生順序の変更について説明する図である。

【図９】PlayList上のマークとClip上のマークについて説明する図である。

【図１０】メニューサムネイルについて説明する図である。

【図１１】PlayListに付加されるマークについて説明する図である。

【図１２】クリップに付加されるマークについて説明する図である。

【図１３】PlayList、Clip、サムネイルファイルの関係について説明する図である。

【図１４】ディレクトリ構造について説明する図である。

【図１５】info.dvrのシンタクスを示す図である。

【図１６】DVR volumeのシンタクスを示す図である。

【図１７】Resumevolumeのシンタクスを示す図である。

【図１８】UIAppInfovolumeのシンタクスを示す図である。

【図１９】Character set valueのテーブルを示す図である。

【図２０】TableOfPlayListのシンタクスを示す図である。

【図２１】TableOfPlayListの他のシンタクスを示す図である。

【図２２】MakersPrivateDataのシンタクスを示す図である。

【図２３】xxxxx.rplsとyyyyy.vplsのシンタクスを示す図である。

【図２４】PlayListについて説明する図である。

【図２５】PlayListのシンタクスを示す図である。

【図２６】PlayList_typeのテーブルを示す図である。

【図２７】UIAppinfoPlayListのシンタクスを示す図である。

【図２８】図２７に示したUIAppinfoPlayListのシンタクス内のフラグについて説明する図である。

【図２９】PlayItemについて説明する図である。

【図３０】PlayItemについて説明する図である。

【図３１】PlayItemについて説明する図である。

【図３２】PlayItemのシンタクスを示す図である。

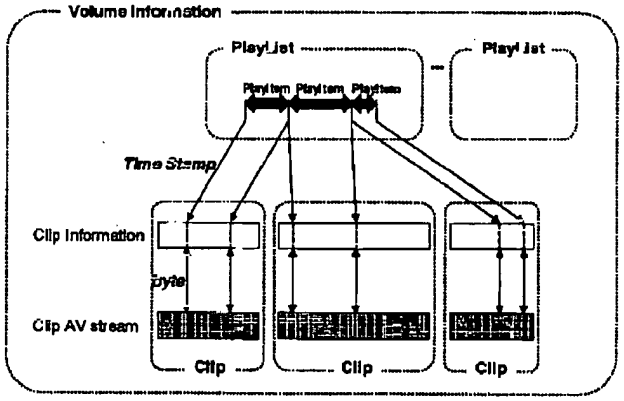
【図３３】IN_timeについて説明する図である。

【図３４】OUT_timeについて説明する図である。

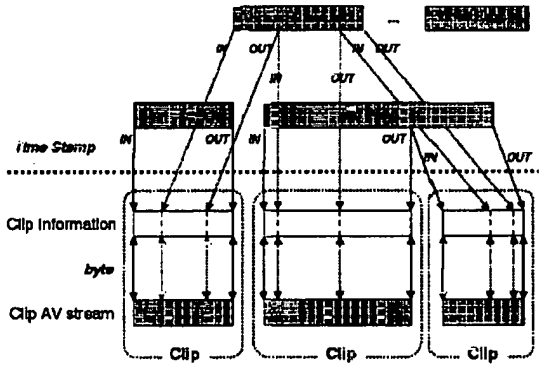
【図35】Connection_Conditionのテーブルを示す図である。
【図36】Connection_Conditionについて説明する図である。
【図37】BridgeSequenceInfoを説明する図である。
【図38】BridgeSequenceInfoのシンタクスを示す図である。
【図39】SubPlayItemについて説明する図である。
【図40】SubPlayItemのシンタクスを示す図である。
【図41】SubPath_typeのテーブルを示す図である。
【図42】PlayListMarkのシンタクスを示す図である。
【図43】Mark_typeのテーブルを示す図である。
【図44】Mark_time_stampを説明する図である。
【図45】zzzzz.clipのシンタクスを示す図である。
【図46】ClipInfoのシンタクスを示す図である。
【図47】Clip_stream_typeのテーブルを示す図である。
【図48】offset_SPNについて説明する図である。
【図49】offset_SPNについて説明する図である。
【図50】ＳＴＣ区間について説明する図である。
【図51】STC_Infoについて説明する図である。
【図52】STC_Infoのシンタクスを示す図である。
【図53】ProgramInfoを説明する図である。
【図54】ProgramInfoのシンタクスを示す図である。
【図55】VideoCondingInfoのシンタクスを示す図である。
【図56】Video_formatのテーブルを示す図である。
【図57】frame_rateのテーブルを示す図である。
【図58】display_aspect_ratioのテーブルを示す図である。
【図59】AudioCondingInfoのシンタクスを示す図である。
【図60】audio_codingのテーブルを示す図である。
【図61】audio_component_typeのテーブルを示す図である。
【図62】sampling_frequencyのテーブルを示す図である。
【図63】CPIについて説明する図である。
【図64】CPIについて説明する図である。
【図65】CPIのシンタクスを示す図である。
【図66】CPI_typeのテーブルを示す図である。
【図67】ビデオEP_mapについて説明する図である。
【図68】EP_mapについて説明する図である。
【図69】EP_mapについて説明する図である。
【図70】EP_mapのシンタクスを示す図である。
【図71】EP_type valuesのテーブルを示す図である。
【図72】EP_map_for_one_stream_PIDのシンタクスを示す図である。
【図73】TU_mapについて説明する図である。
【図74】TU_mapのシンタクスを示す図である。
【図75】ClipMarkのシンタクスを示す図である。
【図76】mark_typeのテーブルを示す図である。
【図77】mark_type_stampのテーブルを示す図である。
【図78】menu.thmbとmark.thmbのシンタクスを示す図である。
【図79】Thumbnailのシンタクスを示す図である。
【図80】thumbnail_picture_formatのテーブルを示す図である。
【図81】tn_blockについて説明する図である。
【図82】DVR MPEG2のトランスポートストリームの構造について説明する図である。
【図83】DVR MPEG2のトランスポートストリームのレコーダモデルを示す図である。
【図84】DVR MPEG2のトランスポートストリームのプレーヤモデルを示す図である。
【図85】source packetのシンタクスを示す図である。
【図86】TP_extra_headerのシンタクスを示す図である。
【図87】copy permission indicatorのテーブルを示す図である。
【図88】シームレス接続について説明する図である。
【図89】シームレス接続について説明する図である。
【図90】シームレス接続について説明する図である
【図91】シームレス接続について説明する図である。
【図92】シームレス接続について説明する図である
【図93】オーディオのオーバーラップについて説明する図である。
【図94】BridgeSequenceを用いたシームレス接続について説明する図である。
【図95】BridgeSequenceを用いないシームレス接続について説明する図である。
【図96】DVR STDモデルを示す図である。
【図97】復号、表示のタイミングチャートを示す図である。
【図98】図1のAVエンコーダの動作を説明する図である。
【図99】ビデオを可変ビットレート符号化して、AVストリームを記録する動作を説明するフローチャートである。
【図100】Video Buffering Verifierを説明する図である。
【図101】VBV制御を説明する図である。
【図102】VBV制御を説明する図である。
【図103】可変ビットレートを制御する場合の例を示す図である。
【図104】可変ビットレート制御の場合の例を示す図である。
【図105】図99のステップＳ21の詳細を説明する

フローチャートである。
【図106】図106のステップS205の詳細を説明するフローチャートである。
【図107】AVストリームの時間経過とAVストリームのデータバイト量との関係を説明する図である。
【図108】ビデオを可変ビットレート符号化して、AVストリームを記録する動作を説明するフローチャートである。
【図109】図108のステップS400の詳細を説明するフローチャートである。
【図110】AVストリームの時間経過とAVストリームのデータバイト量との関係が、比例することを保証する符号化モードを説明するフローチャートである。
【図111】ミニマイズのオペレーションの例を示す図である。
【図112】ミニマイズの時にIN_timeの前の不要なストリームデータを消去する例を示す図である。
【図113】ミニマイズの時にOUT_timeの後ろの不要なストリームデータを消去する例を示す図である。
【図114】EP_mapの作成の動作例を示すフローチャートである。
【図115】媒体を説明する図である。
【符号の説明】
1 記録再生装置, 11乃至13 端子, 14 解析部, 15 AVエンコーダ, 16 マルチプレクサ, 17 スイッチ, 18 多重化ストリーム解析部, 19 ソースパケッタイザ, 20 ECC符号化部, 21 変調部, 22 書き込み部, 23 制御部, 24 ユーザインタフェース, 26 デマルチプレクサ, 27 AVデコーダ, 28 読み出し部, 29復調部, 30 ECC復号部, 31 ソースパケッタイザ, 32, 33 端子

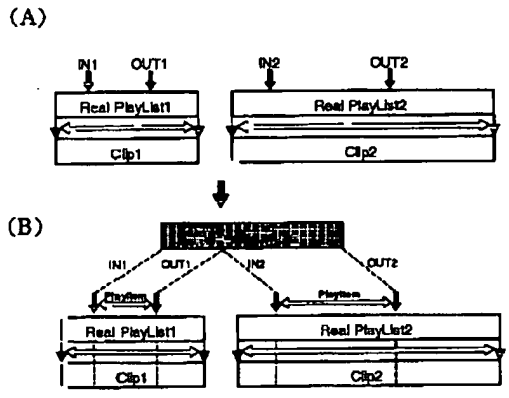
【図2】



【図3】

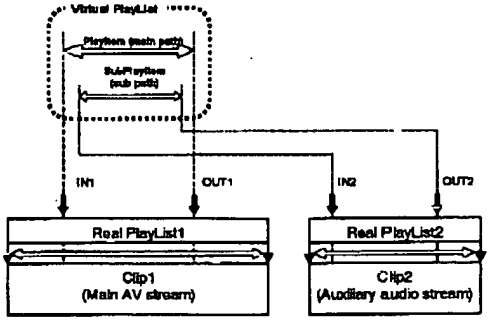


【図6】



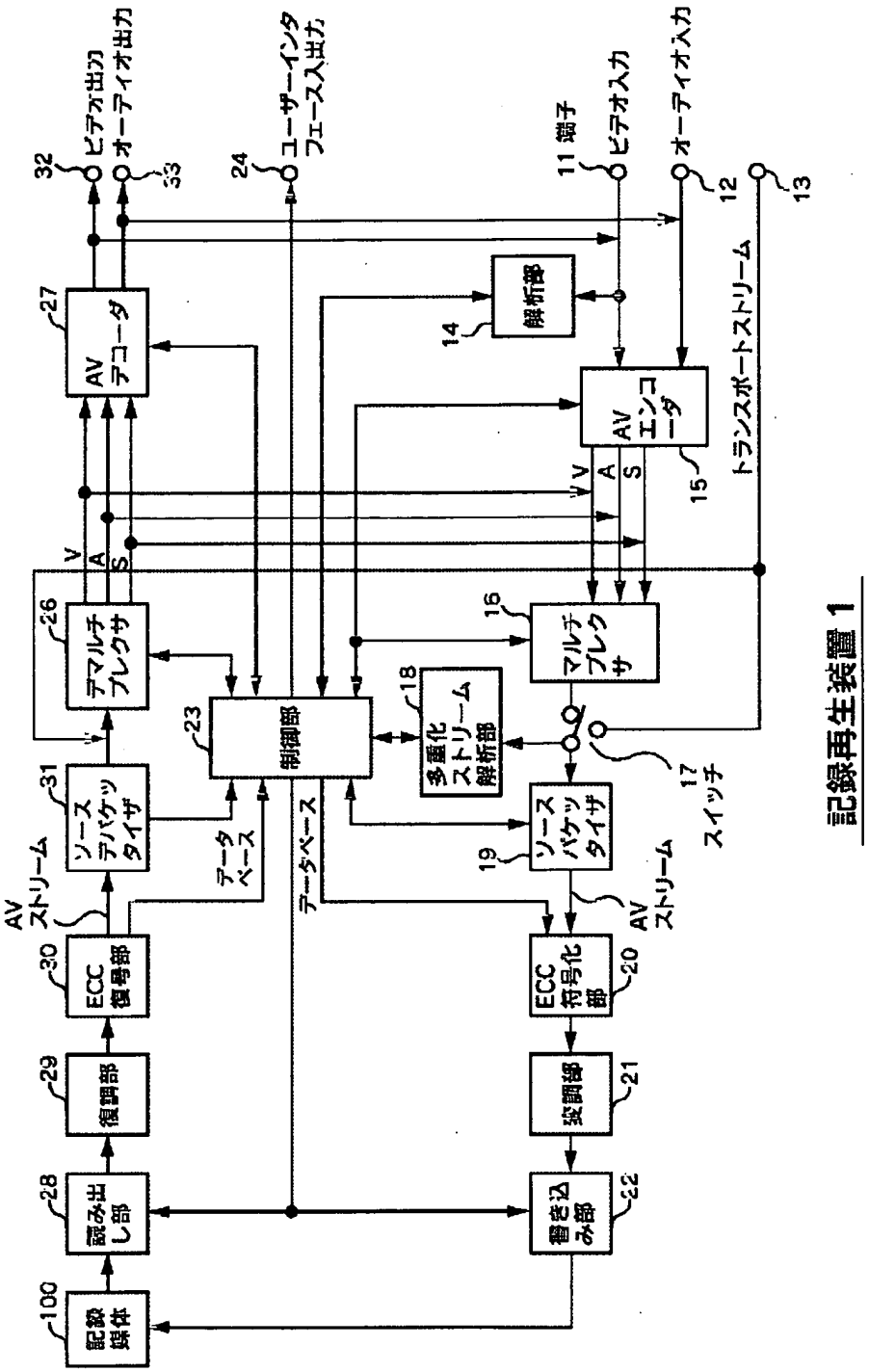
アセンブル編集の例

【図7】



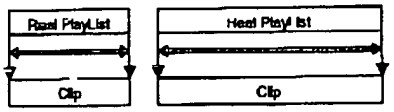
Virtual PlayList へのオーディオのアフレコの例

【図1】


記録再生装置 1
記録媒体
100
読み出し部
28
復調部
29
ECC復号部
30
ソースデパケッタイザ
31
AVストリーム
デマルチプレクサ
26
AVデコーダ
27
ビデオ出力
オーディオ出力
32
33
ユーザーインタフェース入出力
24
制御部
23
データベース
多重化ストリーム解析部
18
V
A
S
解析部
14
AVエンコーダ
15
マルチプレクサ
16
スイッチ
17
ソースパケッタイザ
19
ECC符号化部
20
変調部
21
書き込み部
22
AVストリーム
ビデオ入力
オーディオ入力
11
12
13
端子
トランスポートストリーム

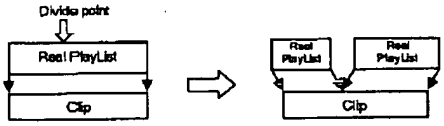
【図4】

(A)



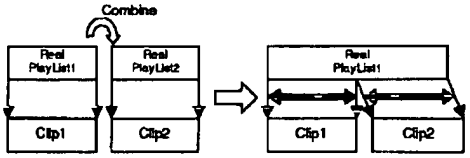
Real PlayListのクリエイトの例

(B)



Real PlayListのディバイドの例

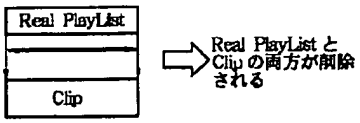
(C)



Real PlayListのコンバインの例

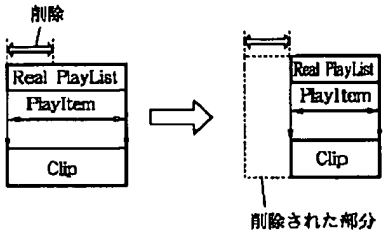
【図5】

(A)



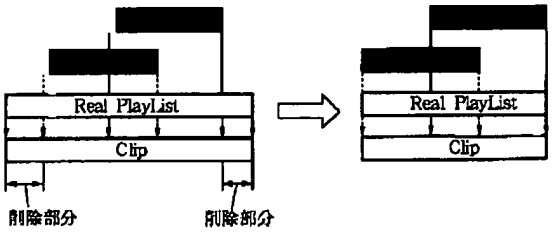
Real PlayList全体のデリートの例

(B)



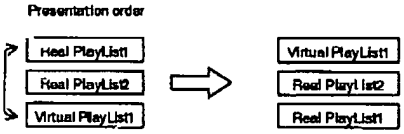
Real PlayListの部分的なデリートの例

(C)



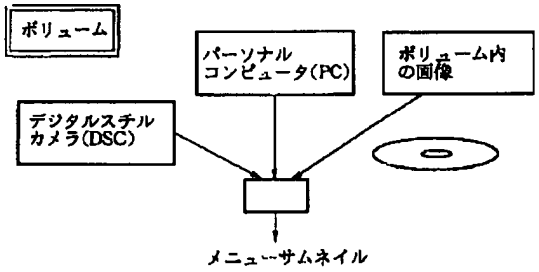
Real PlayListのミニマイズの例

【図8】

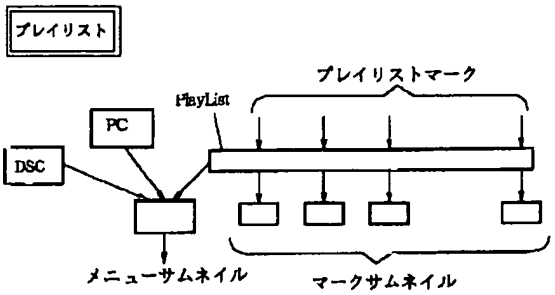


PlayListの再生順序の変更の例

【図10】



【図11】

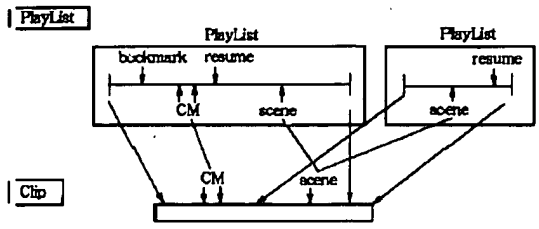


【図19】

Value	Character coding
0x00	Reserved
0x01	ISO/IEC 646 (ASCII)
0x02	ISO/IEC 10646-1 (Unicode)
0x03-0xff	Reserved

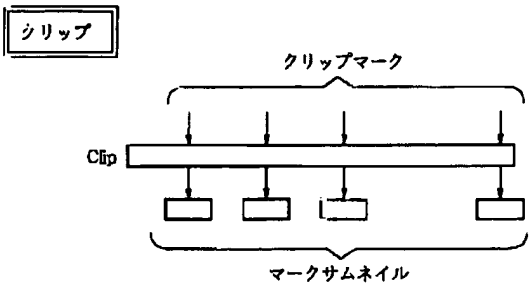
Character set value

【図9】

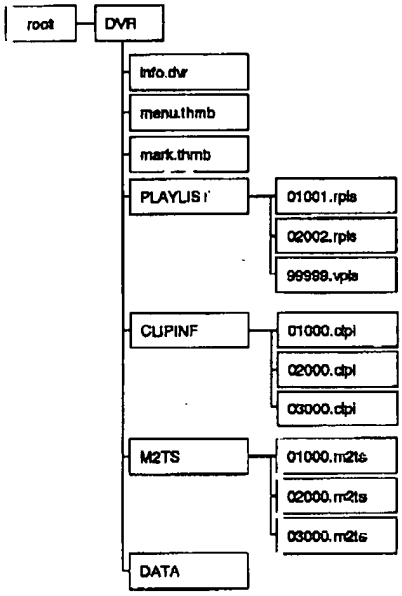


Playlist 上のマークと Clip 上のマーク

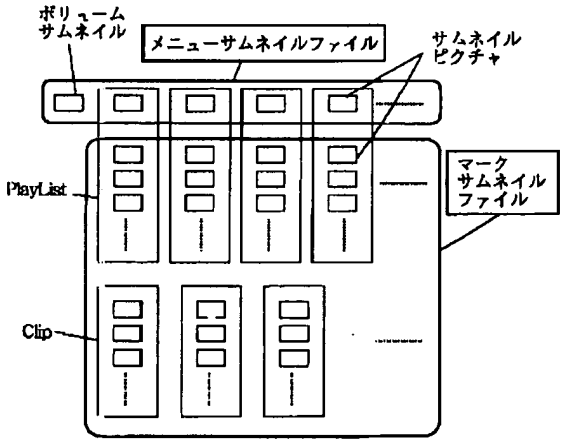
【図12】



【図14】



【図13】



【図15】

Syntax	No. of bits	Mnemonics
info.dvr {		
TableOfPlayLists_Start_address	32	uimsbf
MakerPrivateData_Start_address	32	uimsbf
reserved	192	bslbf
DVRVolume()		
for(i=0; i<N1; i++){		
padding_word	16	bslbf
}		
TableOfPlayLists()		
for(i=0; i<N2; i++){		
padding_word	16	bslbf
}		
MakerPrivateData()		
}		

info.drv のシンタクス

【図66】

CPI_type	Meaning
0	EP_map type
1	TU_map type

CPI_type の意味

【図16】

Syntax	No. of bits	Mnemonics
DVRVolume() {		
version_number	8*4	bslbf
length	32	uimsbf
ResumeVolume()		
UIAppInfoVolume()		
}		

DVR Volume のシンタクス

【図17】

Syntax	No. of bits	Mnemonics
ResumeVolume() {		
reserved	15	bslbf
valid_flag	1	bslbf
resume_PlayList_name	8*10	bslbf
}		

ResumeVolume のシンタクス

【図18】

Syntax	No. of bits	Mnemonics
UIAppInfoVolume () {		
character_set	8	bslbf
name_length	8	uimsbf
Volume_name	8*256	bslbf
reserved	15	bslbf
Volume_protect_flag	1	bslbf
PIN	8*4	bslbf
ref_thumbnail_index	16	uimsbf
reserved_for_future_use	256	bslbf
}		

UIAppInfoVolume のシンタクス

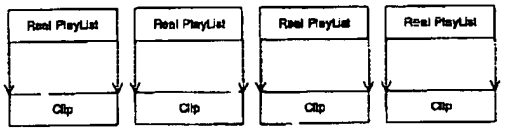
【図20】

Syntax	No. of bits	Mnemonics
TableOfPlayLists() {		
version_number	8*4	bslbf
length	32	uimsbf
number_of_PlayLists	16	uimsbf
for (i=0; i<number_of_PlayLists; i++) {		
PlayList_file_name	8*10	bslbf
}		
}		

TableOfPlayLists のシンタクス

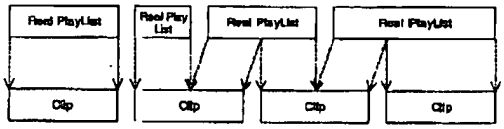
【図24】

(A)



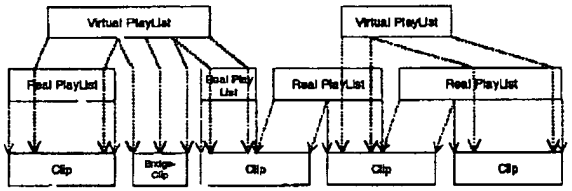
初めて AV ストリームが Clip として記録された時の Real PlayList の例

(B)



編集後の Real PlayList の例

(C)



Virtual PlayList の例

【図26】

PlayList_type	Meaning
0	AV記録のための PlayList この PlayList に参照されるすべての Clip は、一つ以上のビデオストリームを含まなければならない。
1	オーディオ記録のための PlayList この PlayList に参照されるすべての Clip は、一つ以上のオーディオストリームを含まなければならない、そしてビデオストリームを含んではならない。
2 - 255	reserved

PlayList_type

【図41】

SubPath_type	Meaning
0x00	Auxiliary audio stream path
0x01 - 0xff	reserved

SubPath_type

【図21】

■ TableOfPlayLists・シンタクス (4.2.3.2 の別案)

Syntax	No. of bits	Mnemonics
TableOfPlayLists() {		
version_number	8*4	bslbf
length	32	uimsbf
number_of_PlayLists	16	uimsbf
for (i=0; i<*number_of_PlayLists*; i++) {		
PlayList_file_name	8*10	bslbf
UIAppInfoPlayList()		
}		
}		

TableOfPlayLists の別シンタクス

【図22】

Syntax	No. of bits	Mnemonics
MakersPrivateData() {		
version_number	8*4	bslbf
length	32	uimsbf
if(length !=0){		
mpd_blocks_start_address	32	uimsbf
number_of_maker_entries	16	uimsbf
mpd_block_size	16	uimsbf
number_of_mpd_blocks	16	uimsbf
reserved	16	bslbf
for (i=0; i<*number_of_maker_entries*; i++){		
maker_ID	16	uimsbf
maker_model_code	16	uimsbf
start_mpd_block_number	16	uimsbf
reserved	16	bslbf
mpd_length	32	uimsbf
}		
stuffing_bytes	8*2*L1	bslbf
for (i=0; i<*number_of_mpd_blocks*; i++) {		
mpd_block	mpd_block_size*1024*8	
}		
}		
}		

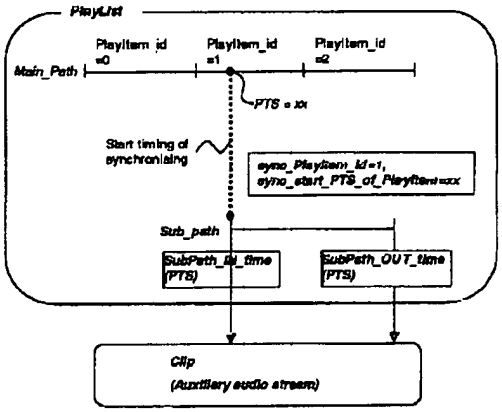
MakersPrivateData のシンタクス

【図23】

Syntax	No. of bits	Mnemonics
xxxxx.rpls / yyyyy.vpls {		
PlayListMark_Start_address	32	uimsbf
MakerPrivateData_Start_address	32	uimsbf
reserved	192	bslbf
PlayList()		
for(i=0; i<N1; i++){		
padding_word	16	bslbf
}		
PlayListMark()		
for(i=0; i<N2; i++){		
padding_word	16	bslbf
}		
MakerPrivateData()		
}		

xxxxx.rpls と yyyyy.vpls のシンタクス

【図39】



【図25】

Syntax	No. of bits	Mnemonics
PlayList() {		
version_number	8*4	bslbf
length	32	uimsbf
PlayList_type	8	uimsbf
CPI_type	1	bslbf
reserved	7	bslbf
UIAppInfoPlayList()		
number_of_PlayItems // main path	16	uimsbf
if (<Virtual PlayList>) {		
number_of_SubPlayItems // sub path	16	uimsbf
}else{		
reserved	16	bslbf
}		
for (PlayItem_id=0; PlayItem_id<number_of_PlayItems; PlayItem_id++) {		
PlayItem() // main path		
}		
if (<Virtual PlayList>) {		
if (CPI_type==0 && PlayList_type==0) {		
for (i = 0; i < number_of_SubPlayItems; i++)		
SubPlayItem() // sub path		
}		
}		
}		

PlayList のシンタクス

【図27】

Syntax	No. of bits	Mnemonics
UIAppInfoPlayList2() {		
character_set	8	bslbf
name_length	8	uimsbf
PlayList_name	8*256	bslbf
reserved	8	bslbf
record_time_and_date	4*14	bslbf
reserved	8	bslbf
duration	4*8	bslbf
valid_period	4*8	bslbf
maker_id	16	uimsbf
maker_code	16	uimsbf
reserved	11	bslbf
playback_control_flag	1	bslbf
write_protect_flag	1	bslbf
is_played_flag	1	bslbf
archive	2	bslbf
ref_thumbnail_index	16	uimsbf
reserved_for_future_use	256	bslbf
}		

UIAppInfoPlayList のシンタクス

【図33】

CPI_type in the PlayList()	Semantics of IN_time
EP_map type	IN_time は、PlayItem の中で最初のプレゼンテーションユニットに対応する33ビット長のPTSの上位32ビットを示さなければならない。
TU_map type	IN_time は、*TU_map_time_axis* 上の時刻でなければならない。かつ、IN_time は、time_unit の精度に丸めて表さねばならない。IN_time は、次に示す等式により計算される。 $IN\_time = TU\_start\_time \% 2^{32}$

IN_time

【図47】

Clip_stream_type	meaning
0	Clip AV ストリーム
1	Bridge-Clip AV ストリーム
2 - 255	Reserved

Clip_stream_type

【図28】

(A)

write_protect_flag	Meaning
0b	その PlayList を自由に消去しても良い。
1b	write_protect_flag を除いてその PlayList の内容は、消去および変更されるべきではない。

write_protect_flag

(B)

is_played_flag	Meaning
0b	その PlayList は、記録されてから一度も再生されたことがない。
1b	PlayList は、記録されてから一度は再生された。

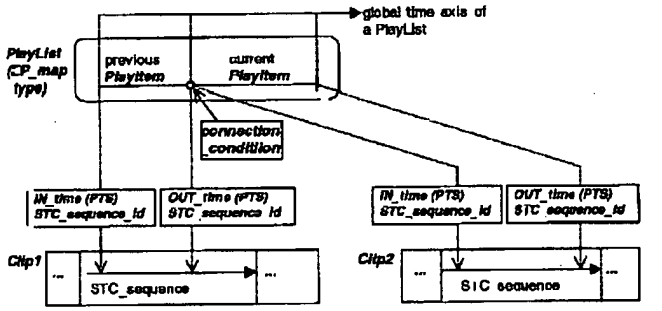
is_played_flag

(C)

archive	Meaning
00b	何も情報が定義されていない。
01b	オリジナル
10b	コピー
11b	reserved

archive

【図29】



PlayList が EP_map type であり、かつ PlayItem が BridgeSequence を持たない時の例

【図32】

Syntax	No. of bits	Mnemonics
PlayItem() {		
Clip_information_file_name	8*10	bslbf
reserved	24	bslbf
STC_sequence_id	8	uimsbf
IN_time	32	uimsbf
OUT_time	32	uimsbf
reserved	14	bslbf
connection_condition	2	bslbf
if (<Virtual PlayList>) {		
if (connection_condition=='10') {		
BridgeSequenceInfo()		
}		
}		
}		

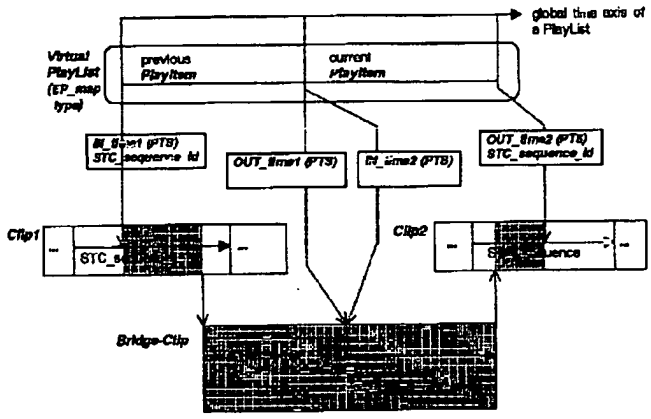
PlayItem のシンタクス

【図34】

CPI_type in the PlayList()	Semantics of OUT_time
EP_map type	OUT_time は、次に示す等式によって計算される Presentation_end_TS の値の上位 32 ビットを示さなければならない。 *Presentation_end_TS = PTS_out + AU_duration* ここで、 *PTS_out* は、PlayItem の中で最後のプレゼンテーションユニットに対応する 33 ビット長の PTS である。 *AU_duration* は、最後のプレゼンテーションユニットの 90kHz 単位の表示期間である。
TU_map type	OUT_time は、*TU_map_time_axis* 上の時刻でなければならない。かつ、OUT_time は、time_unit の精度に丸めて表さねばならない。OUT_time は、次に示す等式により計算される。 *OUT_time = TU_start_time % 2^{32}*

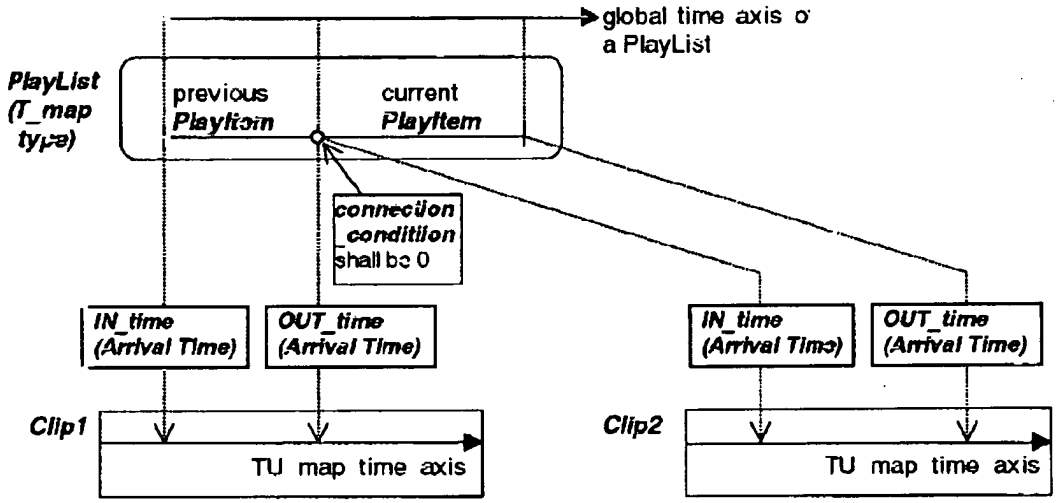
OUT_time

【図30】



PlayList が EP_map type であり、かつ PlayItem が BridgeSequence を持つ時の例

【図31】



PlayList が TU_map type である時の例

【図38】

Syntax	No. of bits	Mnemonics
BridgeSequenceInfo() {		
Bridge_Clip_Information_file_name	8*10	bslbf
RSPN_exit_from_previous_Clip	32	uimsbf
RSPN_enter_to_current_Clip	32	uimsbf
}		

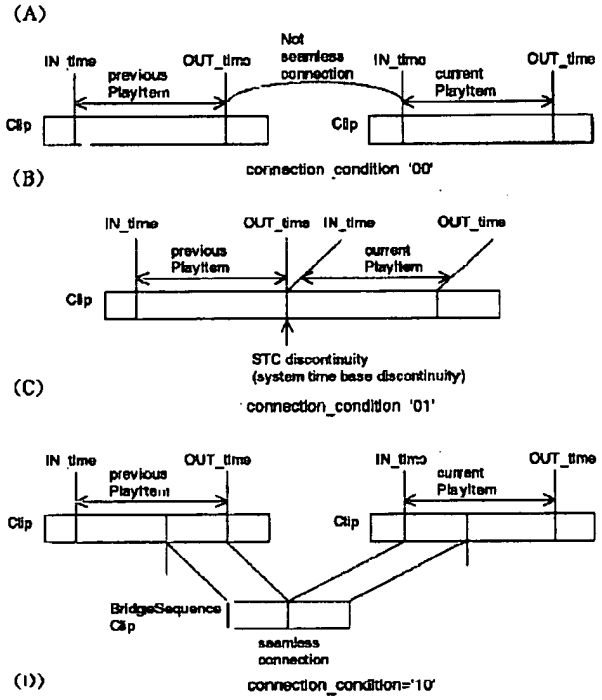
BridgeSequenceInfo のシンタクス

【図35】

connection_condition	meaning
00	• 先行する PlayItem と現在の PlayItem の接続は、シームレス再生の保証がなされていない。 • PlayList の CPI_type が TU map type である場合、connection_condition は、この値をセットされねばならない。
01	• この状態は、PlayList の CPI_type が EP_map type である場合にだけ許される。 • 先行する PlayItem と現在の PlayItem は、システムタイムベース（STC ベース）の不連続点があるために分割されていることを表す。
10	• この状態は、PlayList の CPI_type が EP_map type である場合にだけ許される。 • この状態は、Virtual PlayList に対してだけ許される。 • 先行する PlayItem と現在の PlayItem との接続は、シームレス再生の保証がなされている。 • 先行する PlayItem と現在の PlayItem は、BridgeSequence を使用して接続されており、DVR MPEG-2 トランスポートストリームは、後述する DVR-STD に従っていなければならない。
11	• この状態は、PlayList の CPI_type が EP_map type である場合にだけ許される。 • 先行する PlayItem と現在の PlayItem は、シームレス再生の保証がなされている。 • 先行する PlayItem と現在の PlayItem は、BridgeSequence を使用しないで接続されており、DVR MPEG-2 トランスポートストリームは、後述する DVR-STD に従っていなければならない。

connection__condition

【図36】



connection_condition の説明

【図40】

Syntax	No. of bits	Mnemonics
SubPlayItem() {		
Clip_Information_file_name	8*10	bslbf
SubPath_type	8	bslbf
sync_PlayItem_Id	8	uimsbf
sync_start_PTS_of_PlayItem	32	uimsbf
SubPath_IN_time	32	uimsbf
SubPath_OUT_time	32	uimsbf
}		

SubPlayItem のシンタクス

【図45】

syntax	No. of bits	Mnemonics
zzzzz.clpi {		
STC_Info_Start_address	32	uimsbf
ProgramInfo_Start_address	32	uimsbf
CPI_Start_address	32	uimsbf
ClipMark_Start_address	32	uimsbf
MakerPrivateData_Start_address	32	uimsbf
reserved	96	bslbf
ClipInfo()		
for(i=0; i<N1; i++){		
padding_word	16	bslbf
}		
STC_Info()		
for(i=0; i<N2; i++){		
padding_word	16	bslbf
}		
ProgramInfo()		
for(i=0; i<N3; i++){		
padding_word	16	bslbf
}		
CPI()		
for(i=0; i<N4; i++){		
padding_word	16	bslbf
}		
ClipMark()		
for(i=0; i<N5; i++){		
padding_word	16	bslbf
}		
MakerPrivateData()		
}		

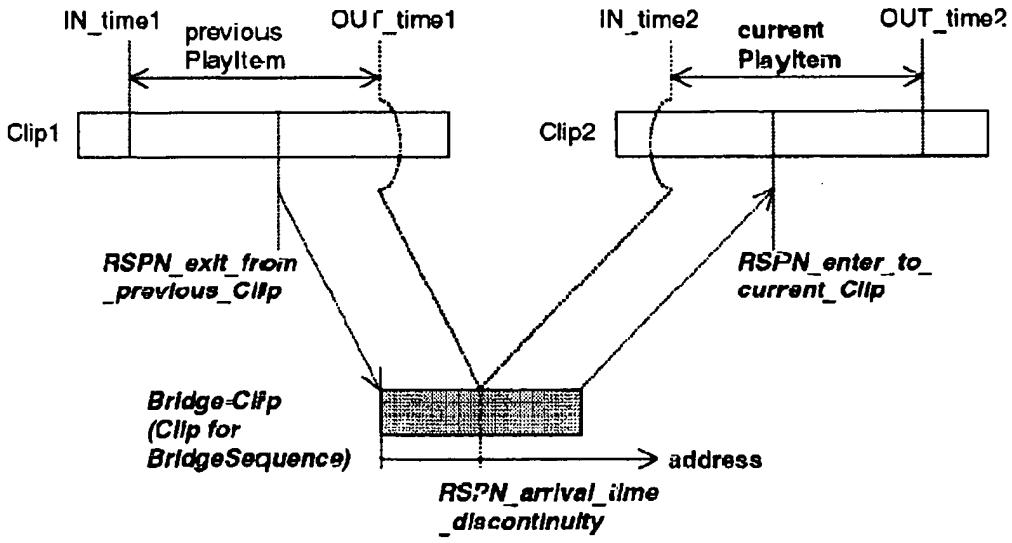
zzzzz.clpi のシンタクス

【図80】

Thumbnail_picture_format	*Meaning*
0x00	*MPEG-2 Video I-picture*
0x01	*DCF (restricted JPEG)*
0x02	*PNG*
0x03-0xff	*reserved*

thumbnail__picture__format

【図37】



【図42】

Syntax	No. of bits	Mnemonics
PlayListMark() {		
version_number	8*4	bslbf
length	32	uimsbf
number_of_PlayList_marks	16	uimsbf
for(i=0; i < number_of_PlayList_marks; i++) {		
reserved	8	bslbf
mark_type	8	bslbf
mark_time_stamp	32	uimsbf
PlayItem_id	8	uimsbf
reserved	24	uimsbf
character_set	8	bslbf
name_length	8	uimsbf
mark_name	8*256	bslbf
ref_thumbnail_index	16	uimsbf
}		
}		

PlayListMark のシンタクス

【図43】

Mark_type	Meaning	Comments
0x00	resume-mark	再生リジュームポイント。PlayListMark()において定義される再生リジュームポイントの数は、0または1でなければならない。
0x01	book-mark	PlayList の再生エントリーポイント。このマークは、ユーザがセットすることができ、例えば、お気に入りのシーンの開始点を指定するマークに使う。
0x02	skip-mark	スキップマークポイント。このポイントからプログラムの最後まで、プレーヤはプログラムをスキップする。PlayListMark() において定義されるスキップマークポイントの数は、0または1でなければならない。
0x03 - 0x8F	reserved	
0x90 - 0xFF	reserved	Reserved for ClipMark()

mark_type

【図44】

CPI_type in the PlayList()	Semantics of mark_time_stamp
EP_map type	mark_time_stamp は、マークで参照されるプレゼンテーションユニットに対応する 33 ビット長の PTS の上位 32 ビットを示さなければならない。
TU_map type	mark_time_stamp は、*TU_map_time_axis* 上の時刻でなければならない。かつ、mark_time_stamp は、time_unit の精度に丸めて表さねばならない。mark_time_stamp は、次に示す等式により計算される。 mark_time_stamp = *TU_start_time* % 2^{32}

mark_time_stamp

【図46】

Syntax	No. of bits	Mnemonics
ClipInfo() {		
version_number	8*4	bslbf
length	32	uimsbf
Clip_stream_type	8	bslbf
offset_SPN	32	uimsbf
TS_recording_rate	24	uimsbf
reserved	8	bslbf
record_time_and_date	4*14	bslbf
reserved	8	bslbf
duration	4*8	bslbf
reserved	7	bslbf
time_controlled_flag	1	bslbf
TS_average_rate	24	uimsbf
if (Clip_stream_type==1) // Bridge-Clip AV stream		
RSPN_arrival_time_discontinuity	32	uimsbf
else		
reserved	32	bslbf
reserved_for_system_use	144	bslbf
reserved	11	bslbf
is_format_identifier_valid	1	bslbf
is_original_network_ID_valid	1	bslbf
is_transport_stream_ID_valid	1	bslbf
is_service_ID_valid	1	bslbf
is_country_code_valid	1	bslbf
format_identifier	32	bslbf
original_network_ID	16	uimsbf
transport_stream_ID	16	uimsbf
service_ID	16	uimsbf
country_code	24	bslbf
stream_format_name	16*8	bslbf
reserved_for_future_use	256	bslbf
}		

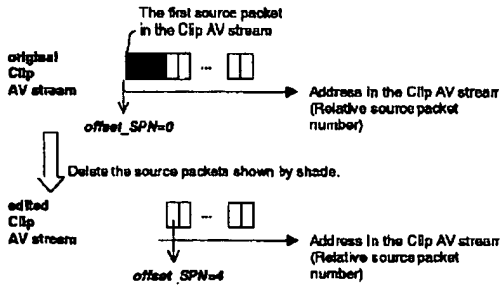
ClipInfo のシンタクス

【図62】

sampling_frequency	Meaning
0	48 kHz
1	44.1 kHz
2	32 kHz
3-254	reserved
255	No information

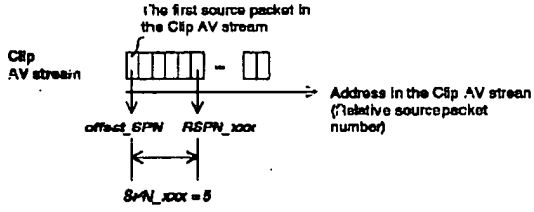
sampling_frequency

【図48】



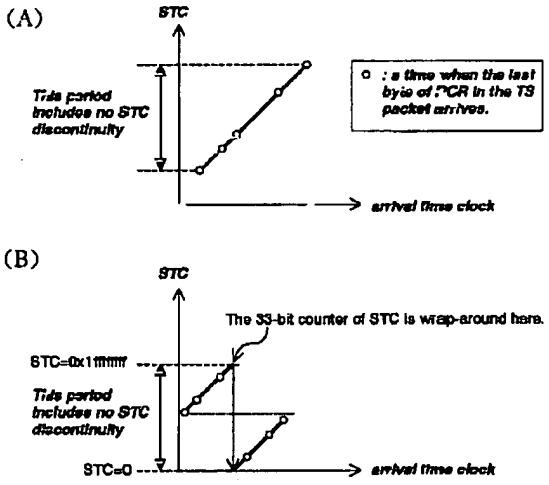
offset_SPN がゼロ以外の値をとる場合の例

【図49】



AV ストリームでの offset_SPN と相対ソースパケット番号（RSPN_xxx）の間の関係

【図50】

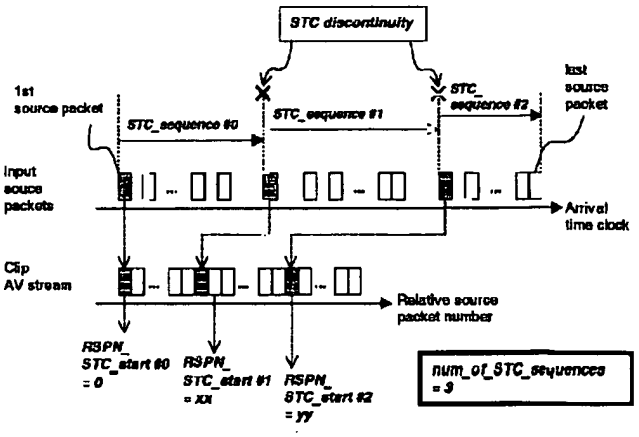


【図56】

video_format	Meaning
0	480i
1	576i
2	480p (including 640x480p format)
3	1080i
4	720p
5	1080p
6 - 254	reserved
255	No information

video_format

【図51】



--STC_Info

【図52】

Syntax	No. of bits	Mnemonics
STC_Info() {		
version_number	8*4	bslbf
length	32	uimsbf
if (length != 0) {		
reserved	8	bslbf
num_of_STC_sequences	8	uimsbf
for(STC_sequence_id=0; STC_sequence_id < num_of_STC_sequences; STC_sequence_id++) {		
reserved	32	bslbf
RSPN_STC_start	32	uimsbf
}		
}		
}		

STC_Infoのシンタクス

【図55】

Syntax	No. of bits	Mnemonics
VideoCodingInfo() {		
video_format	8	uimsbf
frame_rate	8	uimsbf
display_aspect_ratio	8	uimsbf
reserved	8	bslbf
}		

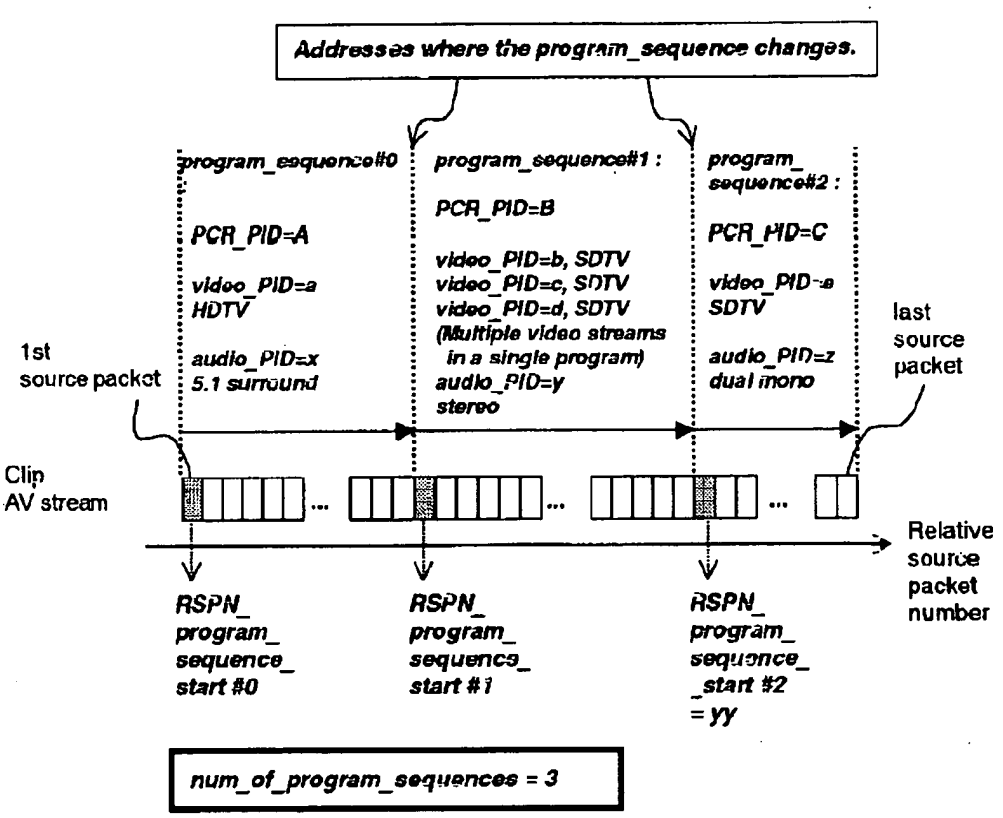
VideoCodingInfoのシンタクス

【図57】

frame_rate	Meaning
0	forbidden
1	24 000/1001 (23.976...)
2	24
3	25
4	30 000/1001 (29.97..)
5	30
6	50
7	60 000/1001 (59.94 ..)
8	60
9 - 254	reserved
255	No information

frame_rate

【図53】



ProgramInfoの例

【図54】

Syntax	No. of bits	Mnemonics
ProgramInfo() {		
length	32	uimsbf
if (length != 0) {		
reserved	8	bslbf
number_of_program_sequences	8	uimsbf
for(i=0; i<number_of_program_sequences; i++){		
RSPN_program_sequence_start	32	uimsbf
reserved	48	bslbf
PCR_PID	16	bslbf
number_of_videos	8	uimsbf
number_of_audios	8	uimsbf
for (k=0; k<number_of_videos; k++) {		
video_stream_PID	16	bslbf
VideoCodingInfo()		
}		
for (k=0; k<number_of_audios; k++) {		
audio_stream_PID	16	bslbf
AudioCodingInfo()		
}		
}		
}		
}		

ProgramInfoのシンタクス

【図60】

audio_coding	Meaning
0	MPEG-1 audio layer I or II
1	Dolby AC-3 audio
2	MPEG-2 AAC
3	MPEG-2 multi-channel audio, backward compatible to MPEG-1
4	SESF LPCM audio
5-254	reserved
255	No information

audio_coding

【図58】

display_aspect_ratio	Meaning
0	forbidden
1	reserved
2	4:3 display aspect ratio
3	16:9 display aspect ratio
4-254	reserved
255	No information

display_aspect_ratio

【図59】

Syntax	No. of bits	Mnemonics
AudioCodingInfo() {		
audio_coding	8	uimsbf
audio_component_type	8	uimsbf
sampling_frequency	8	uimsbf
reserved	8	bslbf
}		

AudioCodingInfo のシンタクス

【図65】

Syntax	No. of bits	Mnemonics
CPI() {		
version_number	8*4	bslbf
length	32	uimsbf
reserved	15	bslbf
CPI_type	1	bslbf
if (CPI_type == 0)		
EP_map()		
else		
TU_map()		
}		

CPI のシンタクス

【図61】

audio_component_type	Meaning
0	single mono channel
1	dual mono channel
2	stereo (2-channel)
3	multi-lingual, multi-channel
4	surround sound
5	audio description for the visually impaired
6	audio for the hard of hearing
7-254	reserved
255	No information

audio_component_type

【図70】

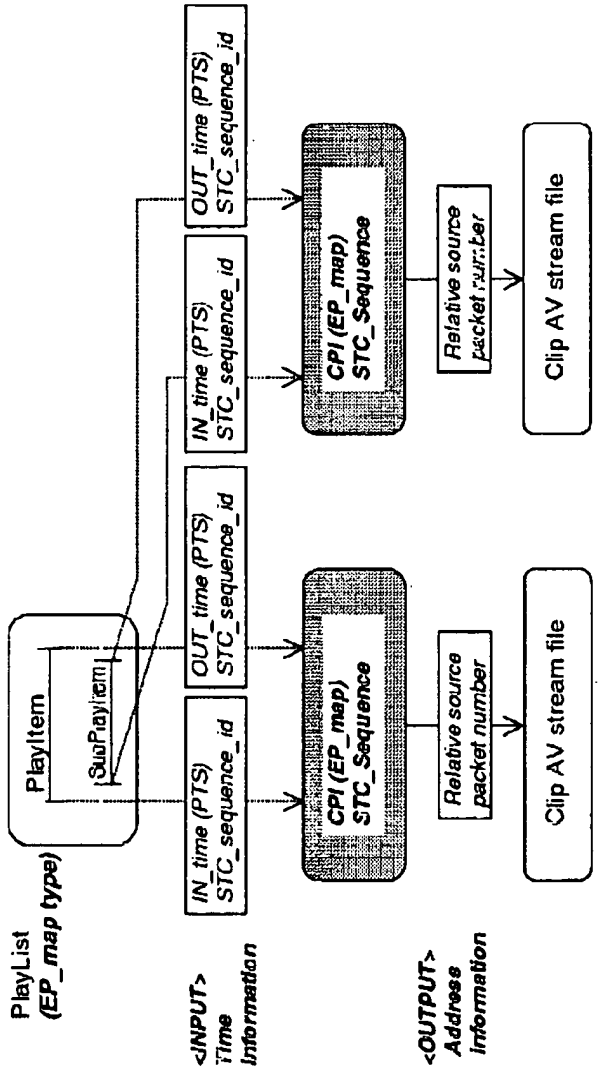
Syntax	No. of bits	Mnemonics
EP_map(){		
reserved	12	bslbf
EP_type	4	uimsbf
number_of_stream_PIDs	16	uimsbf
for (*k=0;k<number_of_stream_PIDs;k++*){		
stream_PID *(k)*	16	bslbf
num_EP_entries *(k)*	32	uimsbf
EP_map_for_one_stream_PID_Start_address *(k)*	32	uimsbf
}		
for(i=0;i<X;i++){		
padding_word	16	bslbf
}		
for (*k=0;k<number_of_stream_PIDs;k++*){		
EP_map_for_one_stream_PID(*num_EP_entries(k)*)		
for(i=0;i<Y;i++){		
padding_word	16	bslbf
}		
}		
}		

【図72】

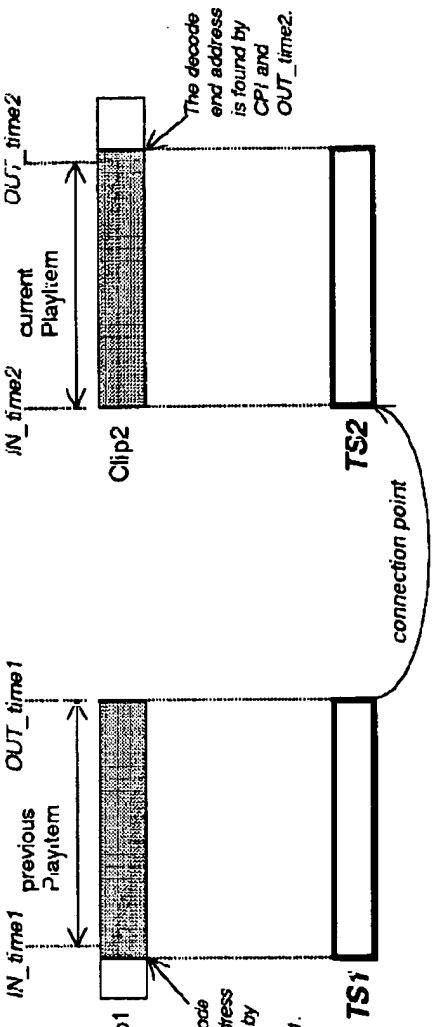
Syntax	No. of bits	Mnemonics
EP_map_for_one_stream_PID(*N*){		
for (i=0; i< *N*; i++) {		
PTS_EP_start	32	uimsbf
RSPN_EP_start	32	uimsbf
}		
}		

EP_map_for_one_stream_PID のシンタクス

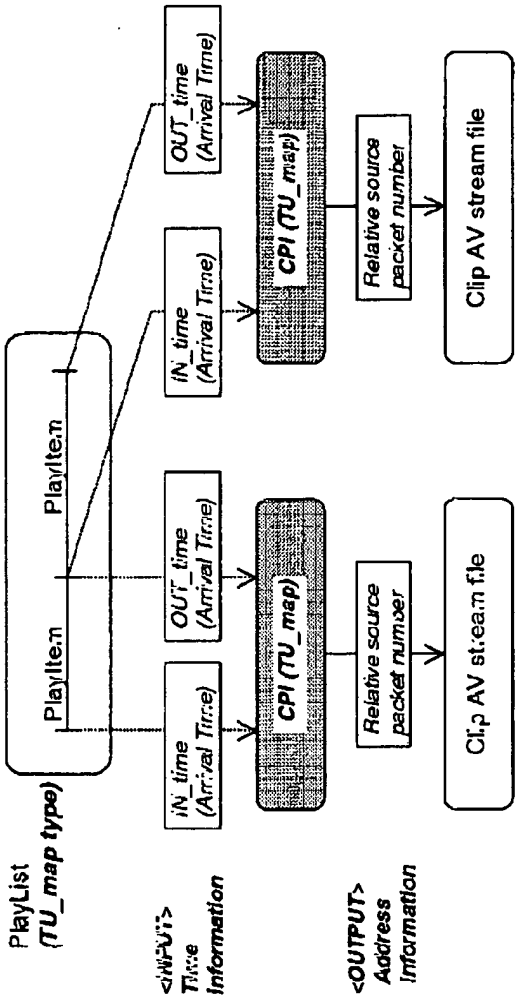
【図63】


PlayList
(EP_map type)
PlayItem
SubPlayItem
<INPUT>
Time
Information
IN_time (PTS)
STC_sequence_id
OUT_time (PTS)
STC_sequence_id
IN_time (PTS)
STC_sequence_id
OUT_time (PTS)
STC_sequence_id
<OUTPUT>
Address
information
CPI (EP_map)
STC_Sequence
CPI (EP_map)
STC_Sequence
Relative source
packet number
Relative source
packet number
Clip AV stream file
Clip AV stream file

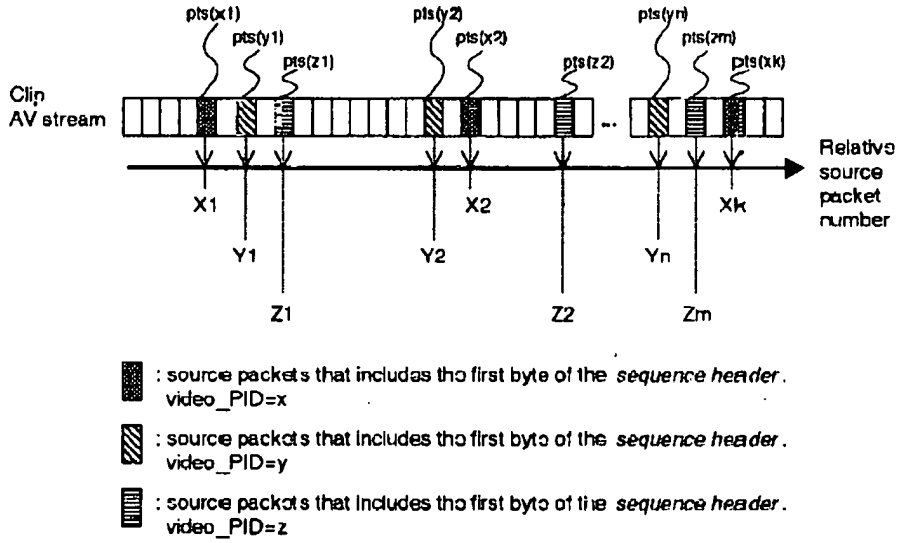
【図89】

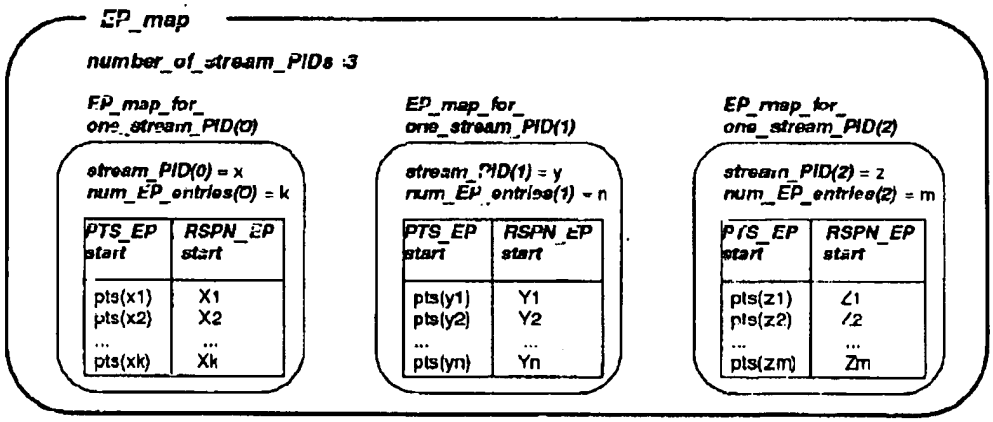

The decode
end address
is found by
CPI and
OUT_time2.
OUT_time2
current
PlayItem
IN_time2
Clip2
TS2
connection point
OUT_time1
previous
PlayItem
IN_time1
Clip1
TS1
The decode
start address
is found by
CPI and
IN_time1.

【図64】


PlayList
(TU_map type)
PlayItem
PlayItem
<INPUT>
Time
Information
IN_time
(Arrival Time)
OUT_time
(Arrival Time)
CPI (TU_map)
<OUTPUT>
Address
Information
Relative source
packet number
Clip AV stream file
IN_time
(Arrival Time)
OUT_time
(Arrival Time)
CPI (TU_map)
Relative source
packet number
Clip AV stream file

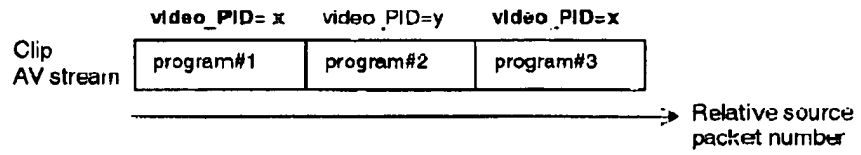
【図67】

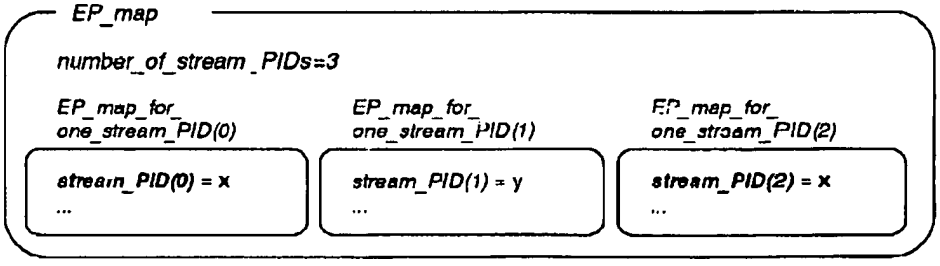




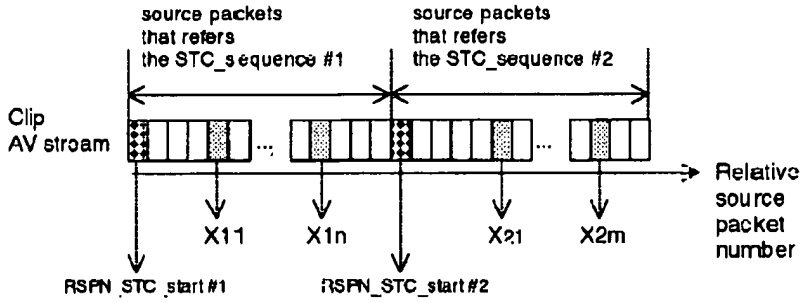
ビデオの EP_map の例

【図69】





【図68】



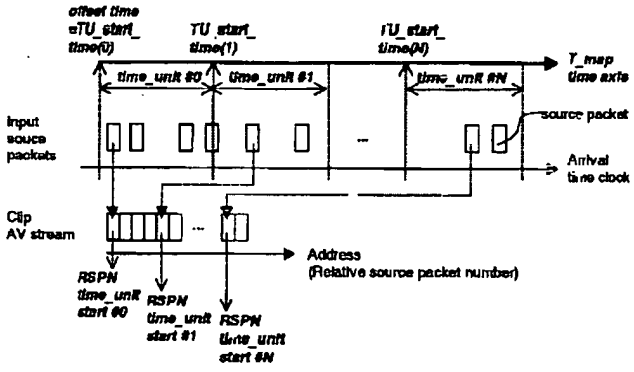
EP_map_for_one_stream_PID

video_PID=x

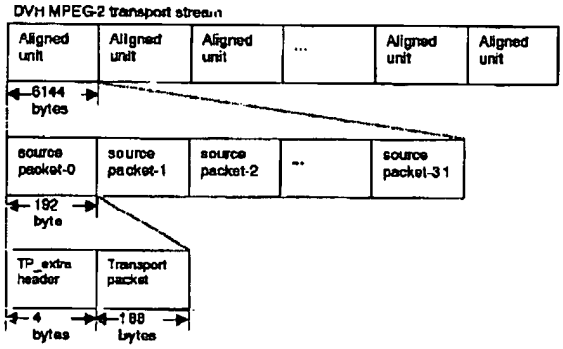
PTS_EP start	*RSPN_EP start*	
pts(x11)	X11	*These data belong to the STC_sequence #1*
...	...	
pts(x1n)	X1n	
		boundary
pts(x21)	X21	*These data belong to the STC_sequence #2*
...	...	
pts(x2m)	X2m	

RSPN_STC_start #2 < X21

【図73】



【図82】



DVR MPEG-2 トランスポートストリームの構造

【図76】

Mark_type	Meaning	Comments
0x00 - 0x8F	reserved	Reserved for PlayListMark()
0x90	Event-start mark	番組の開始ポイントを示すマーク点。
0x91	Local event-start mark	番組の中の局所的な場面を示すマーク点。
0x92	Scene-start mark	シーンチェンジポイントを示すマーク。
0x93 - 0xFF	reserved	

mark_type

【図87】

copy_permission indicator	meaning
00	copy free
01	no more copy
10	copy once
11	copy prohibited

copy permission indicator table

【図71】

EP_type	Meaning
0	video
1	audio
2 - 15	reserved

EP_type Values

【図74】

Syntax	No. of bits	Mnemonics
TU_map(){		
offset_time	32	bslbf
time_unit_size	32	uimsbf
number_of_time_unit_entries	32	uimsbf
for (k=0; k<*number_of_time_unit_entries*; k++)		
RESERVED	32	uimsbf
}		

TU_map のシンタクス

【図77】

CPI_type in the CPI()	Semantics of mark_time_stamp
EP_map type	mark_time_stamp は、マークで参照されるプレゼンテーションユニットに対応する 33 ビット長の PTS の上位 32 ビットを示さなければならない。
TU_map type	mark_time_stamp は、*TU_map_time_axis* 上の時刻でなければならない。かつ、mark_time_stamp は、time_unit の精度に丸めて表さねばならない。mark_time_stamp は、次に示す等式により計算される。 mark_time_stamp = *TU_start_time* % 2^{32}

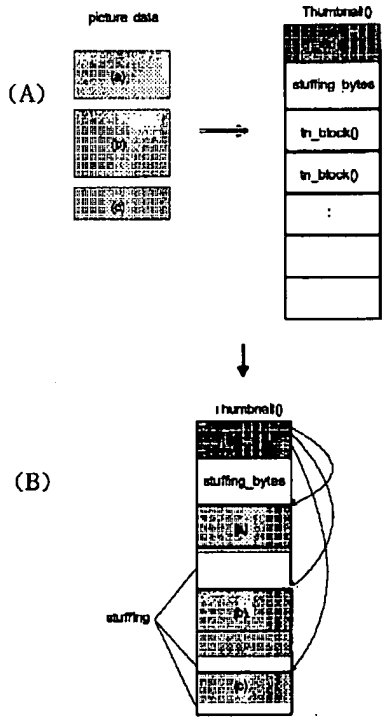
mark_type_stamp

【図75】

Syntax	No. of bits	Mnemonics
ClipMark() {		
version_number	8*4	bslbf
length	32	uimsbf
number_of_Clip_marks	16	uimsbf
for(i=0; i < *number_of_Clip_marks*; i++) {		
reserved	8	bslbf
mark_type	8	bslbf
mark_time_stamp	32	uimsbf
STC_sequence_id	8	uimsbf
reserved	24	bslbf
character_set	8	bslbf
name_length	8	uimsbf
mark_name	8*255	bslbf
ref_thumbnail_index	16	uimsbf
}		
}		

ClipMark のシンタクス

【図81】



【図85】

Syntax	No. of bits	Mnemonics
source_packet() {		
TP_extra_header()		
transport_packet()		
}		

source packet

【図78】

Syntax	No. of bits	Mnemonics
menu.thmb / mark.thmb {		
reserved	256	bslbf
Thumbnail()		
for(i=0; i<N1; i++)		
padding_word	16	bslbf
}		

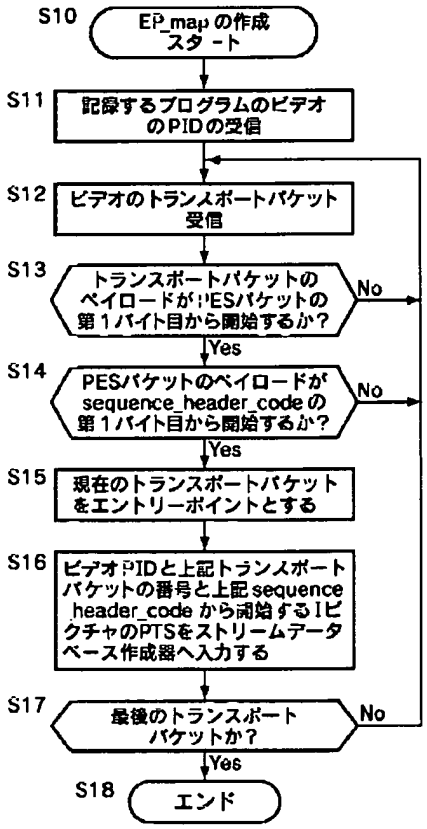
menu.thmb と mark.thmb のシンタクス

【図79】

Syntax	Bits	Mnemonics
Thumbnail() {		
version_number	8*4	char
length	32	uimsbf
if (length != 0) {		
tn_blocks_start_address	32	bslbf
number_of_thumbnails	16	uimsbf
tn_block_size	16	uimsbf
number_of_tn_blocks	16	uimsbf
reserved	16	bslbf
for(i = 0; i < *number_of_thumbnails*; i++) {		
thumbnail_index	16	uimsbf
thumbnail_picture_format	8	bslbf
reserved	8	bslbf
picture_data_size	32	uimsbf
start_tn_block_number	16	uimsbf
x_picture_length	16	uimsbf
y_picture_length	16	uimsbf
reserved	16	uimsbf
}		
stuffing_bytes	8*2*L1	bslbf
for(k = 0; k <*number_of_tn_blocks*; k++) {		
tn_block	tn_block_size* 1024*8	
}		
}		
}		

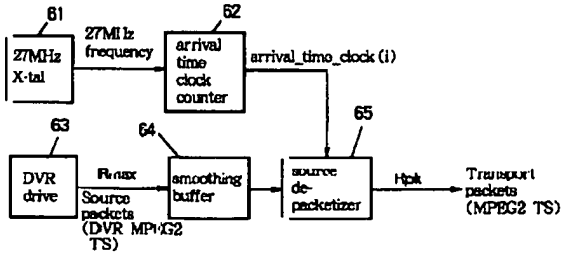
Thumbnail のシンタクス

【図114】



EP_map の作成の動作例を説明するフローチャート

【図84】



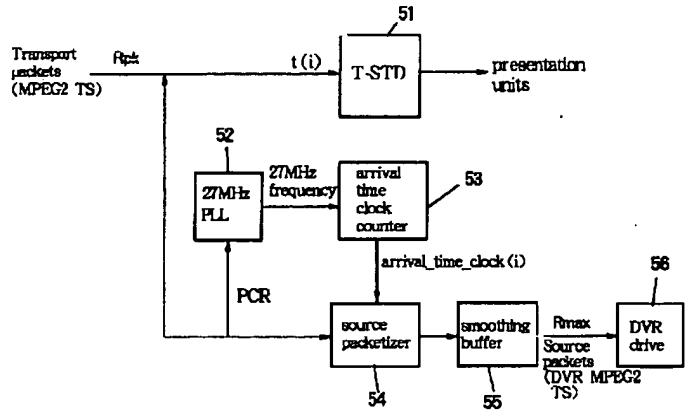
DVR MPEG-2 トランスポートストリームのプレーヤモデル

【図86】

Syntax	No. of bits	Mnemonics
TP_extra_header() {		
copy_permission_indicator	2	uimsbf
arrival_time_stamp	30	uimsbf
}		

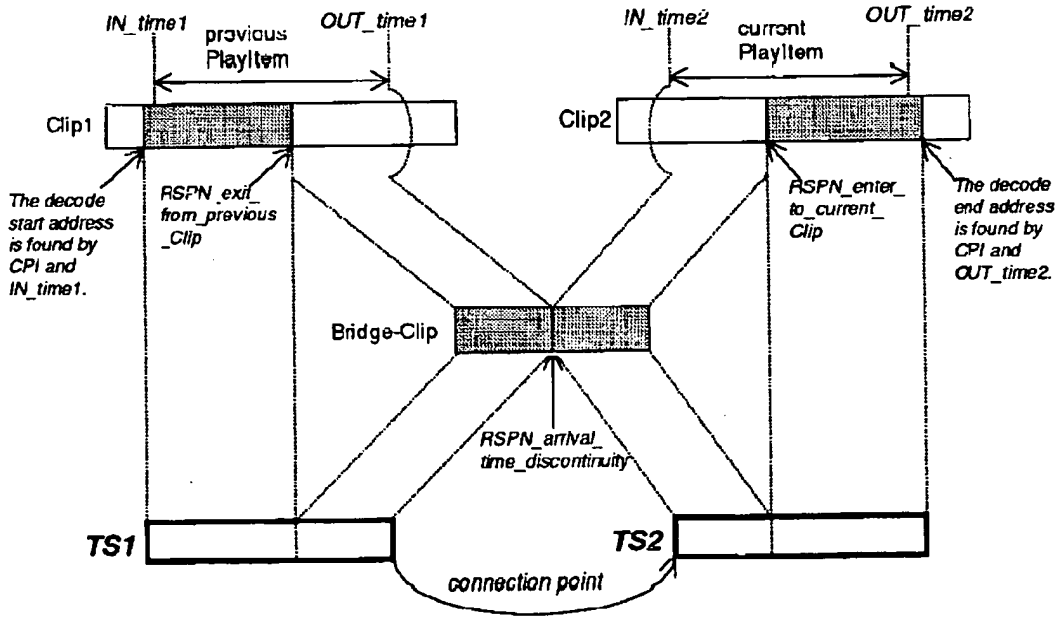
TP_extra_header

【図83】

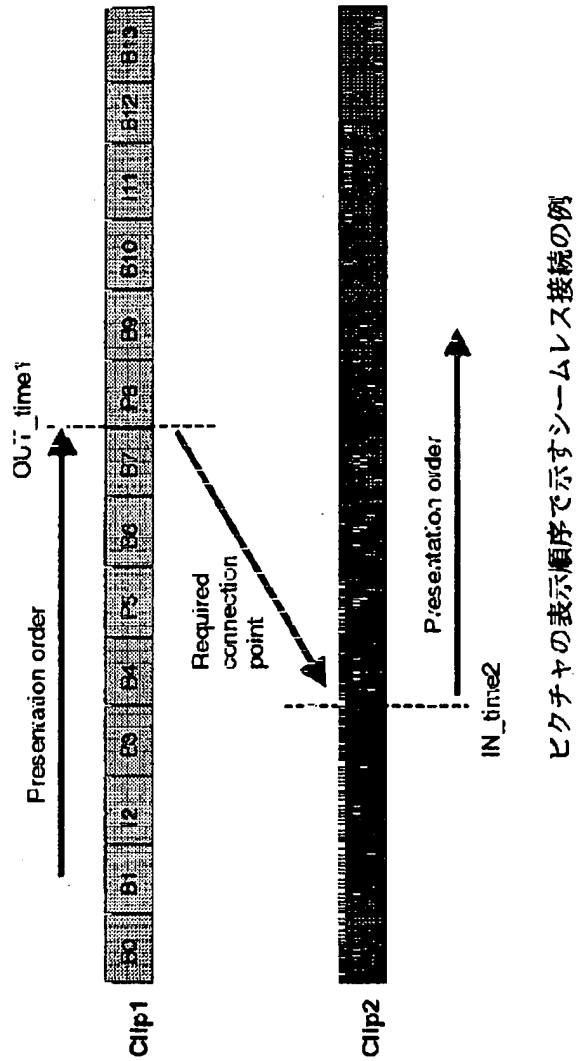


DVR MPEG-2 トランスポートストリームのレコーダモデル

【図88】

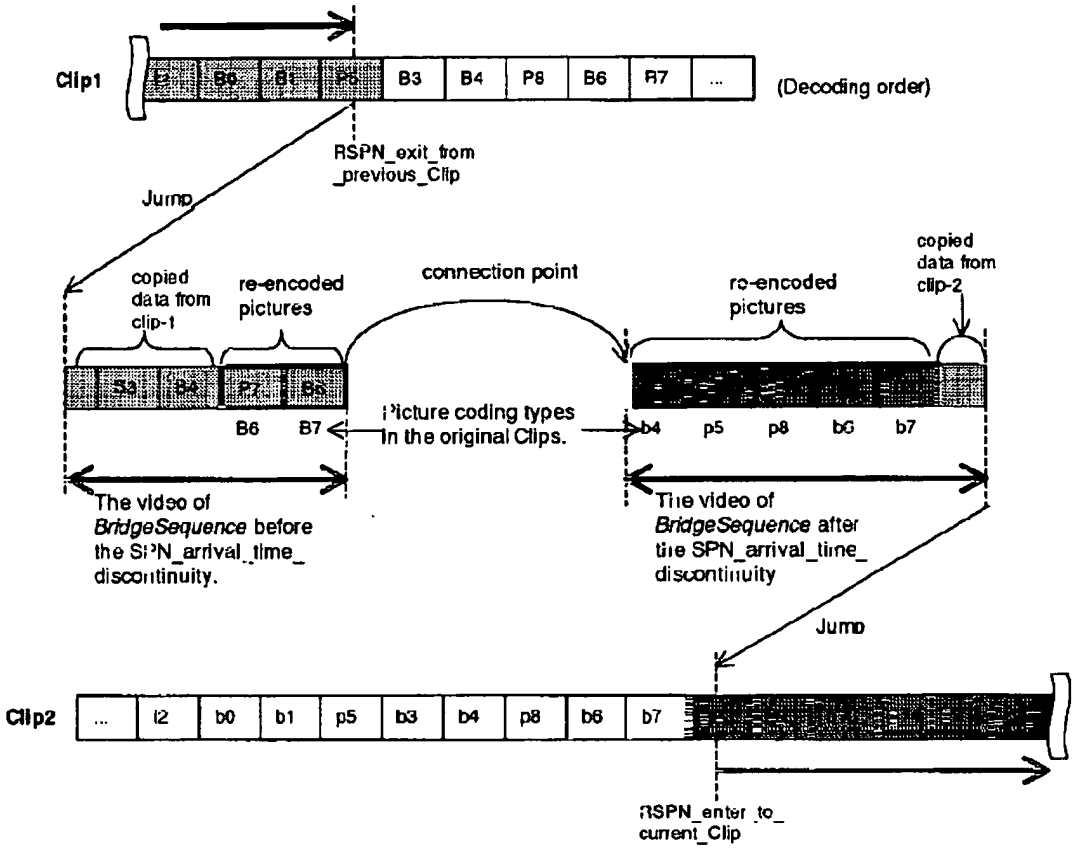


【図90】



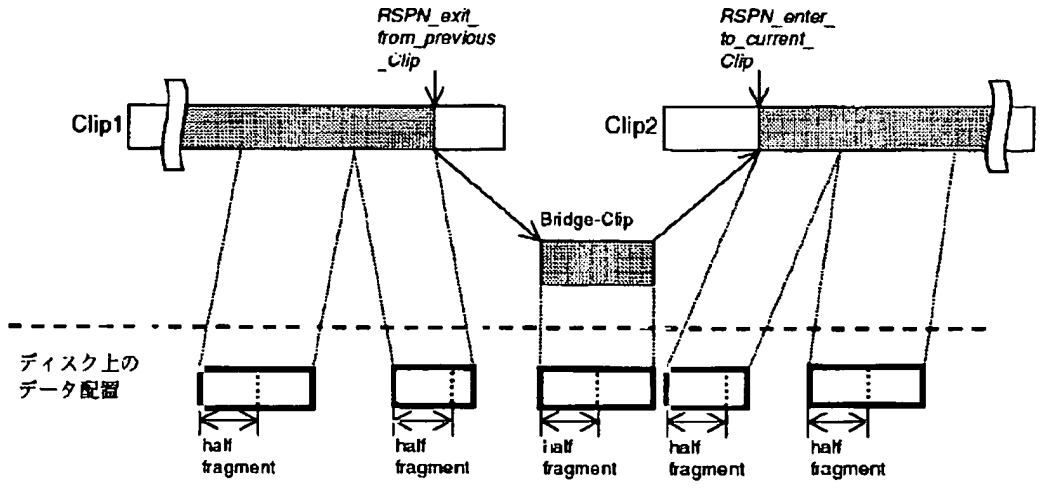
ピクチャの表示順序で示すシームレス接続の例

【図91】



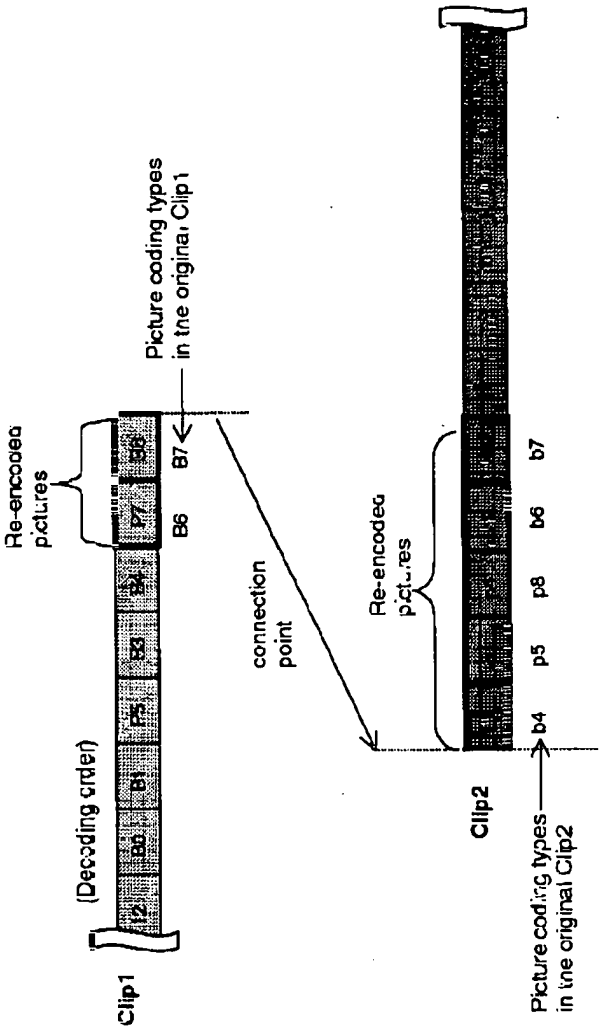
BridgeSequenceを使用してシームレス接続を実現する例1

【図94】



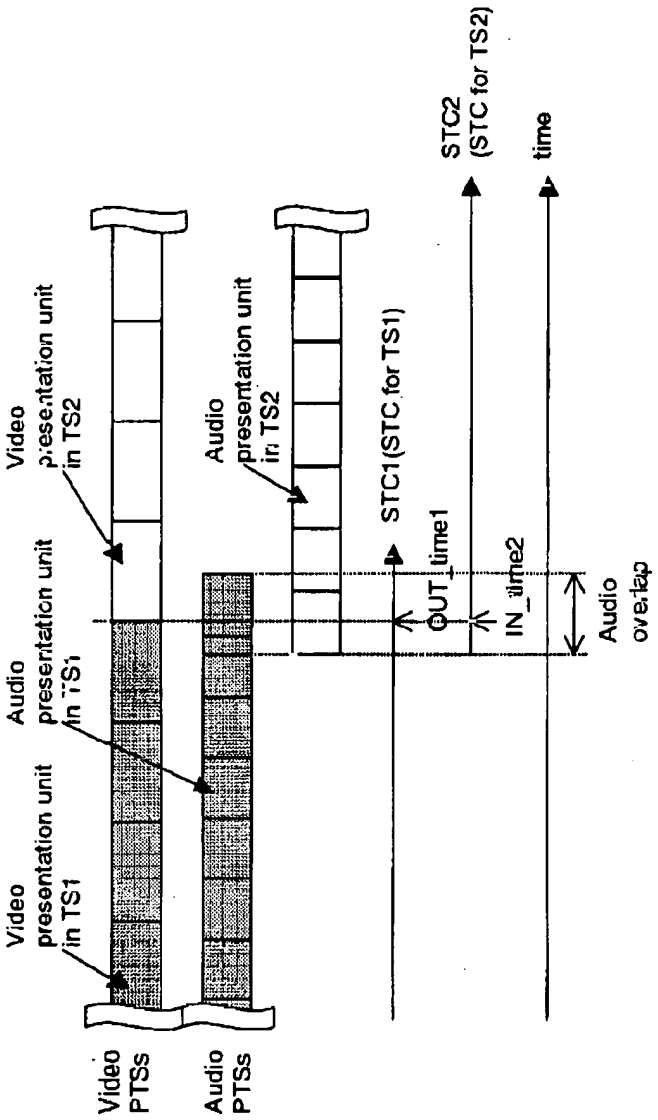
BridgeSequenceを使用してシームレス接続をする場合の、データアロケーションの例

【図92】

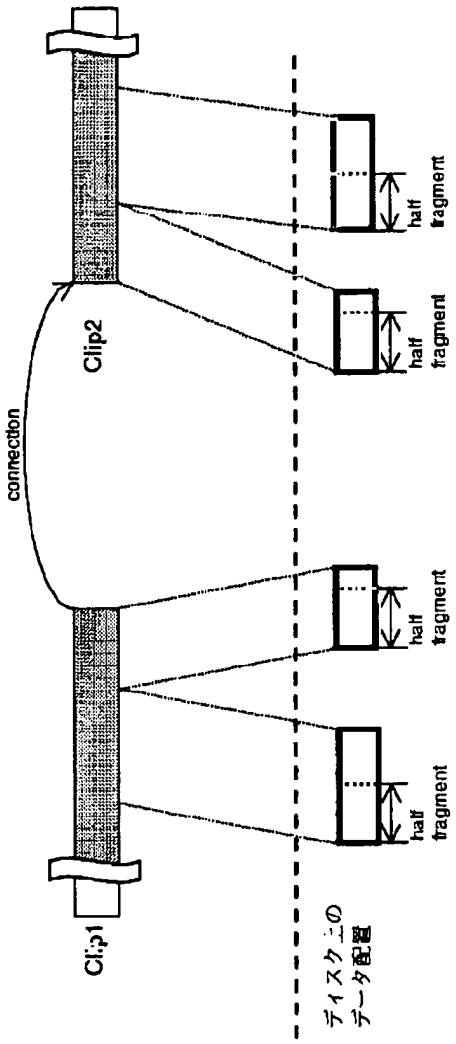


BridgeSequence を使用しないでシームレス接続を実現する例2

【図93】

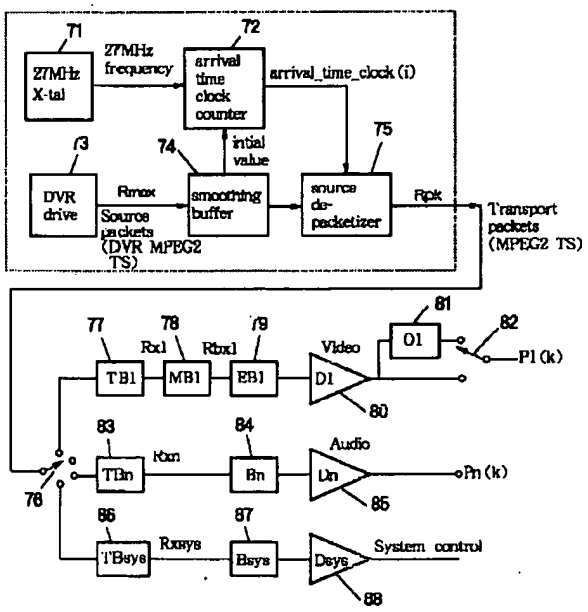

Video presentation unit in TS1
Audio presentation unit in TS1
Video presentation unit in TS2
Audio presentation unit in TS2
Video PTSs
Audio PTSs
STC1(STC for TS1)
STC2 (STC for TS2)
time
OUT_time1
IN_time2
Audio overlap

【図95】

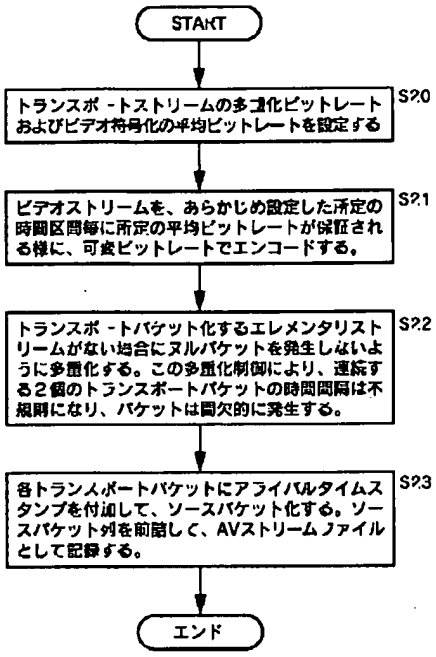


BridgeSequence を使用しないでシームレス接続をする場合の、データアロケーションの例

【図96】

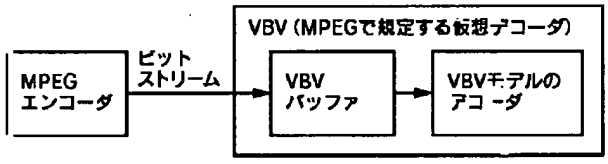


【図99】



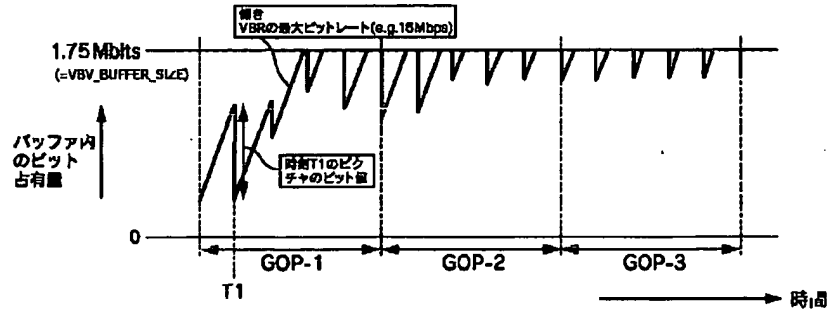
AVストリームが時間経過とAVストリームのデータバイト量との関係が、比例することを保証する符号化モード（time_controlled_flag=1）において、ビデオを可変ビットレート符号化して、AVストリームを記録する動作を説明するフローチャート

【図100】



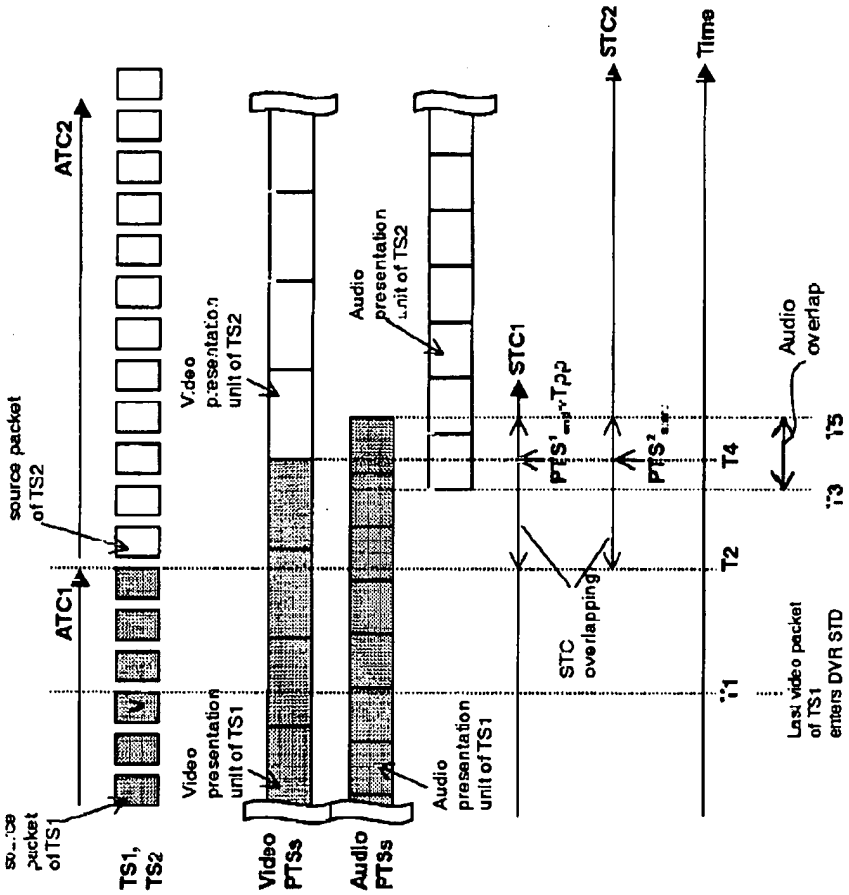
VBV (Video Buffering Verifier) を説明する図

【図101】



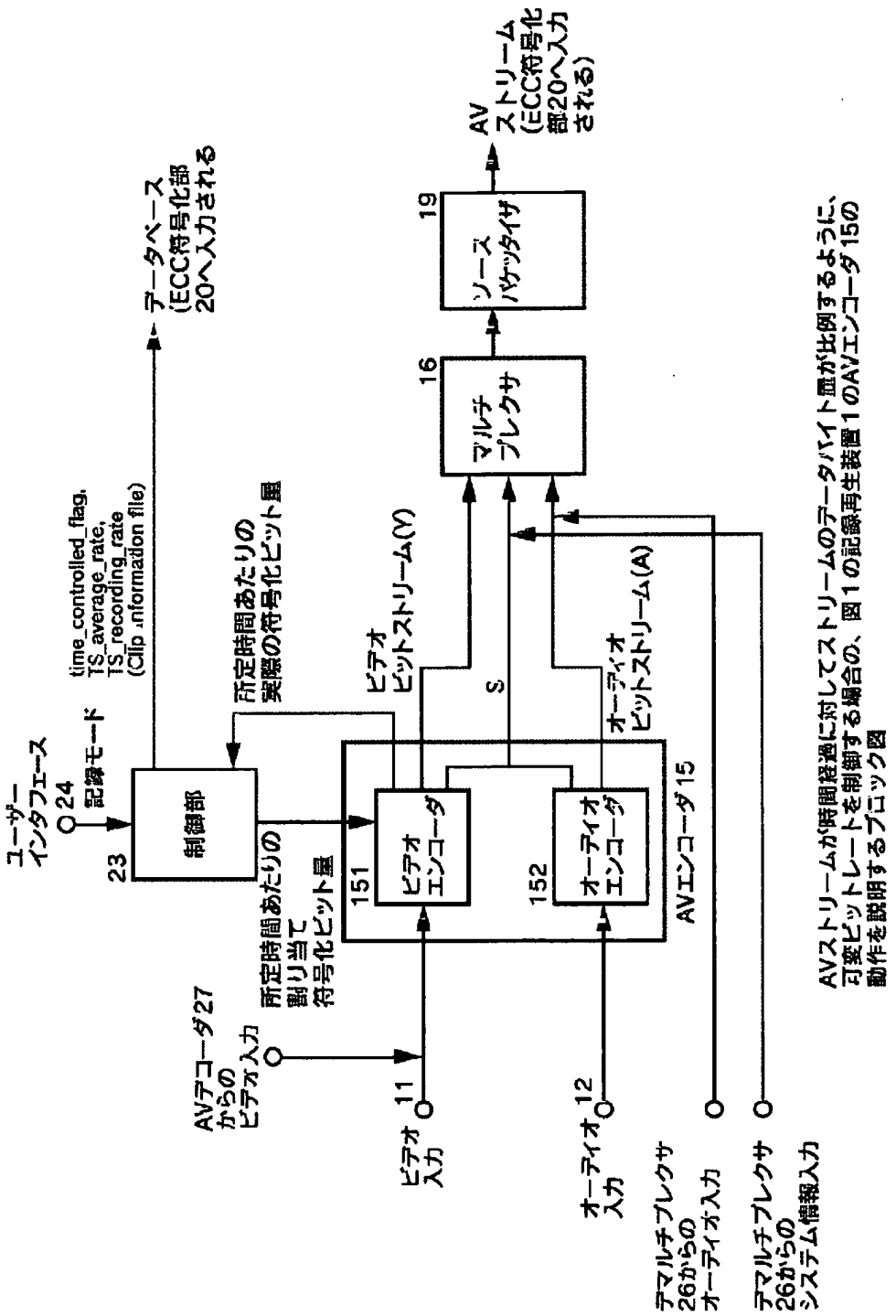
VBVバッファに空きがあるときは、バッファへの入力ビットレートがVBR(Variable Bit-Rate)の最大ビットレートであり、VBVバッファのビット占有量がフルの場合は、バッファへの入力ビットレートをゼロにする場合のVBV制御を説明する図

【図97】



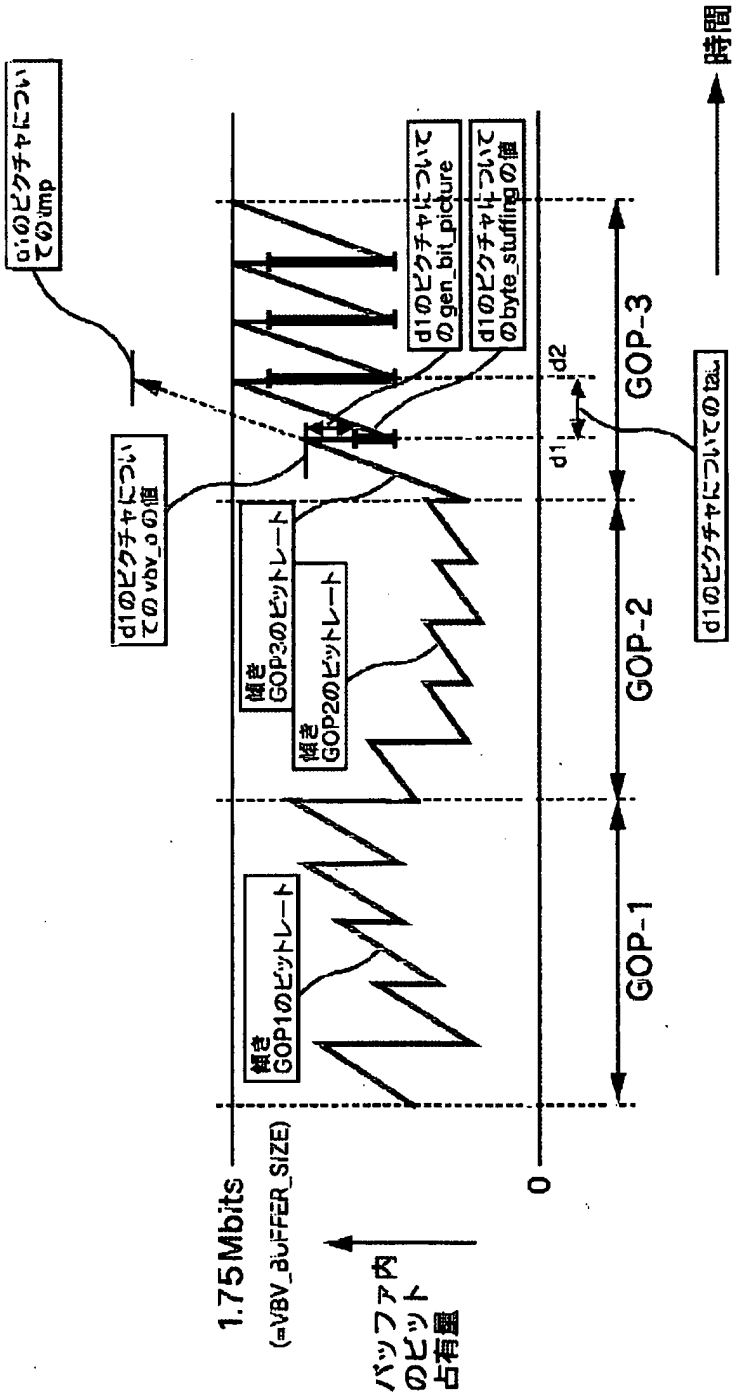
あるAVストリーム(TS1)からそれにシームレスに接続された次のAVストリーム(TS2)へと移る時のトランスポートパケットの入力、復号、表示のタイミングチャート

【図98】



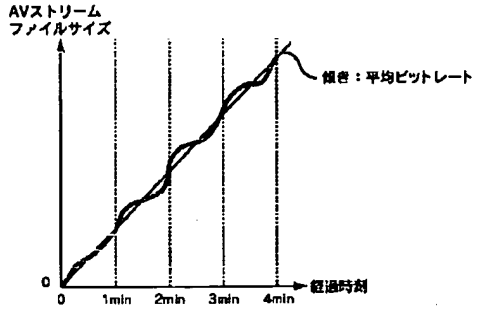
AVストリームが時間経過に対してストリームのデータバイト量が比例するように、可変ビットレートを制御する場合の、図1の記録再生装置1のAVエンコーダ15の動作を説明するブロック図

【図102】



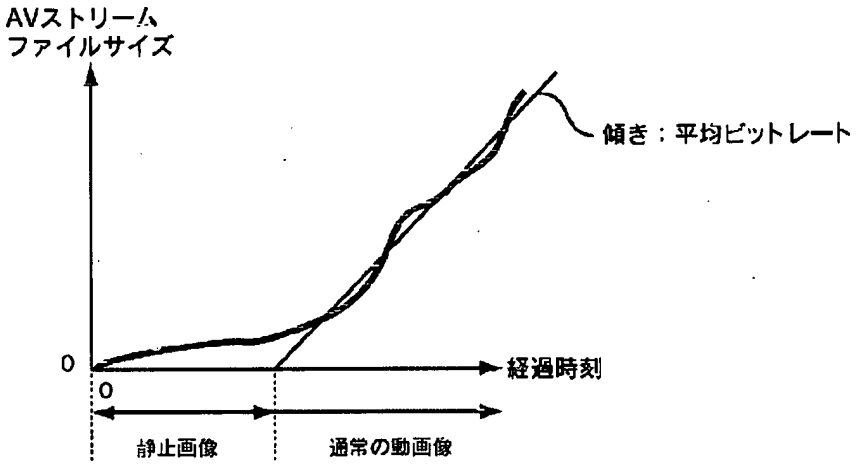
GOP内でCBR (Constant Bit-Rate) の場合のVBV制御
VBVバッファへの入力ビットレートが、現在のGOPの符号化ビットレートであり、VBVバッファがオーバーフローしないようにバイトスタッフィングを挿入する場合のVBV制御を説明する図

【図103】



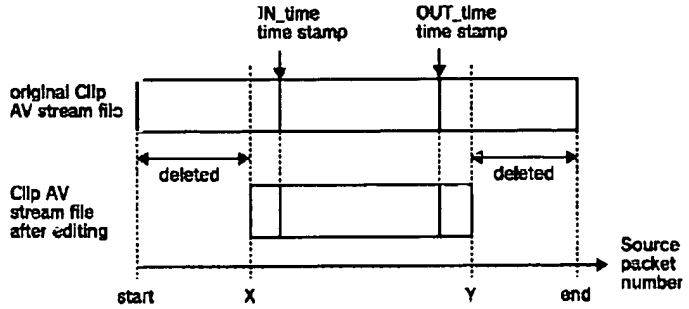
AVストリームの時間経過とAVストリームのデータバイト量との関係が、所定の誤差の範囲以内で比例するように、可変ビットレートを制御する場合の例

【図104】



AVストリームの時間経過とAVストリームのデータバイト量との関係が比例しない場合の、可変ビットレート制御の場合の例（静止画像が数分続いた後に通常の動画像になる場合）

【図111】



ミニマイズのオペレーションの例

【図105】

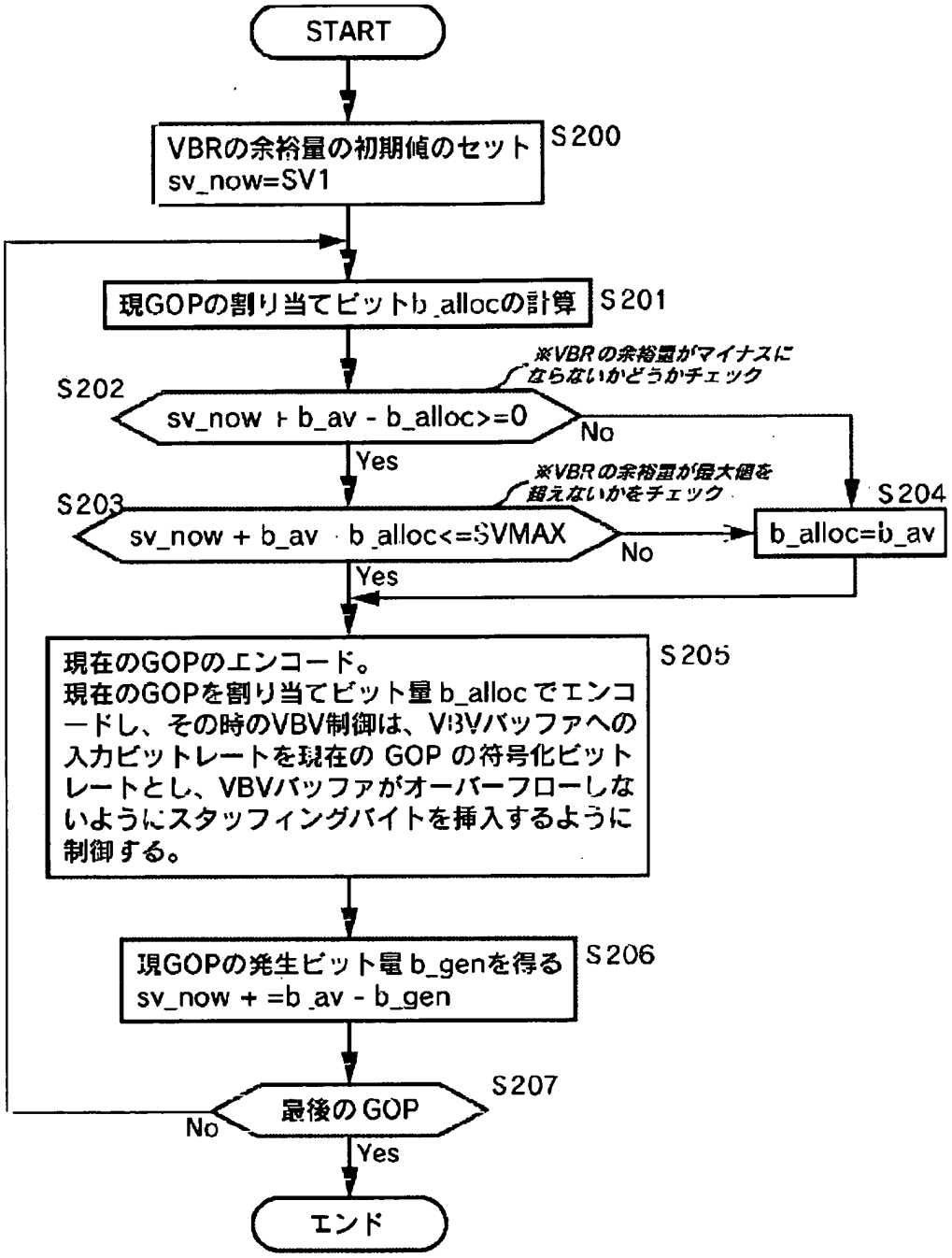


図99 のステップ21の詳細を説明するフローチャート

【図106】

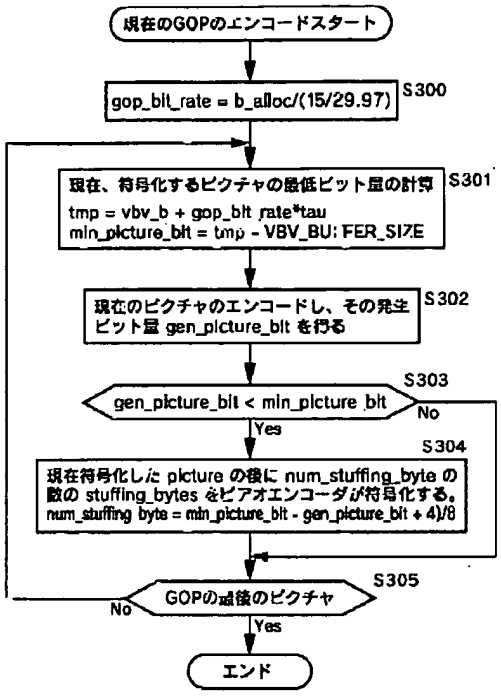
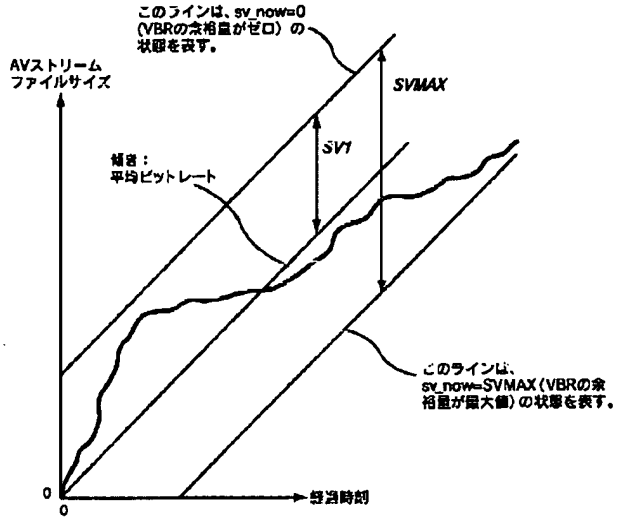


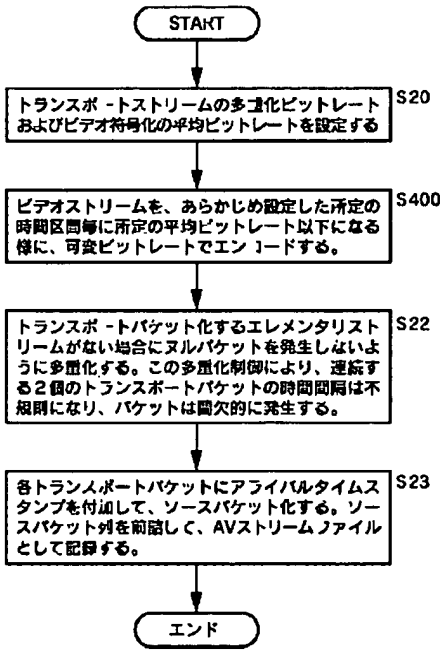
図105のステップS205の詳細を説明するフローチャート

【図107】



VBR (Variable Bit-Rate) 制御をする場合に、VBRの余裕分の最大何と最小値の両方を制限する場合において、AVストリームの時間経過とAVストリームのデータバイト量との関係を説明する図

【図108】



AVストリームが時間経過とAVストリームのデータバイト量との関係が、比例することを保証しない符号化モード（time_controlled_flag :0）において、ビデオを可変ビットレート符号化して、AVストリームを記録する動作を説明するフローチャート

【図109】

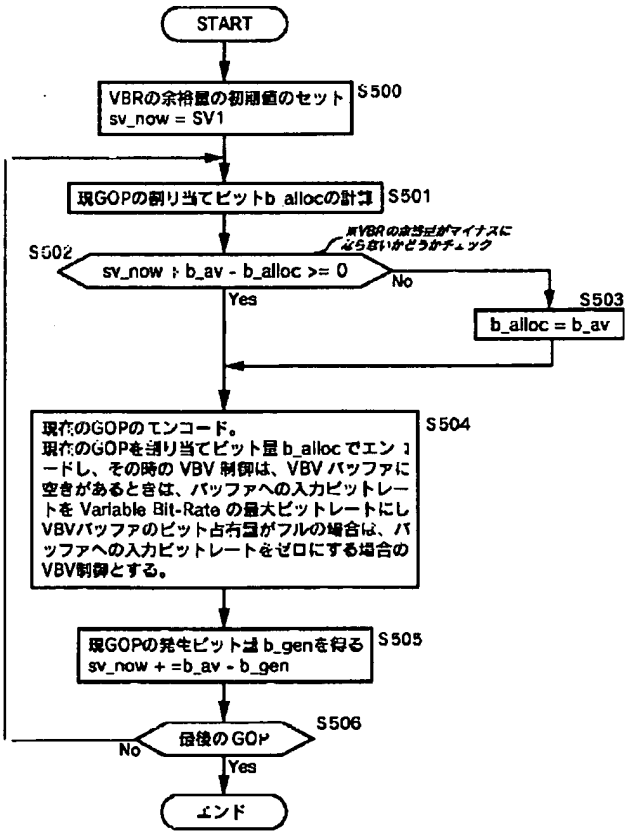


図108のステップ400の詳細を説明するフローチャート

【図110】

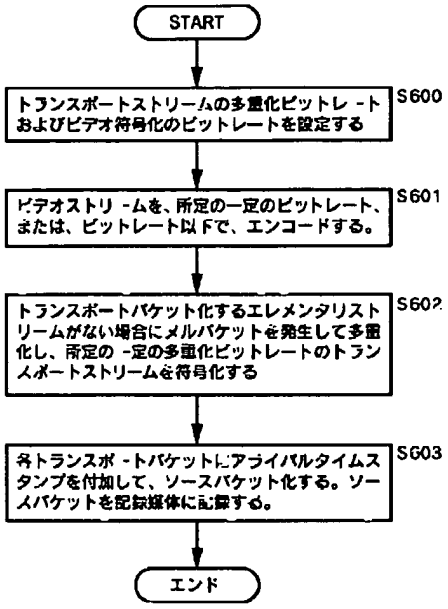
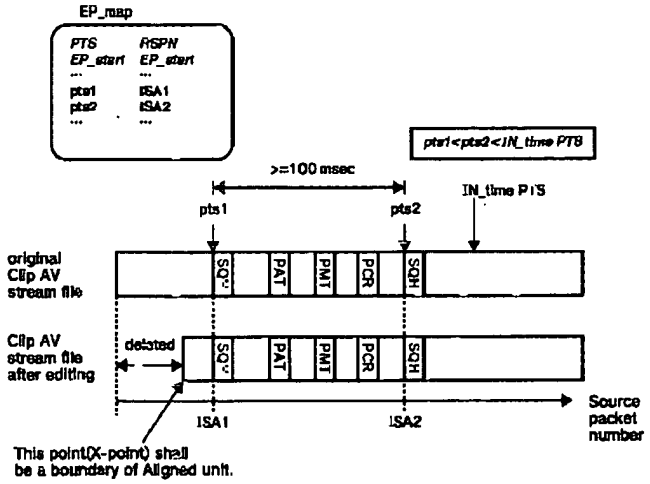


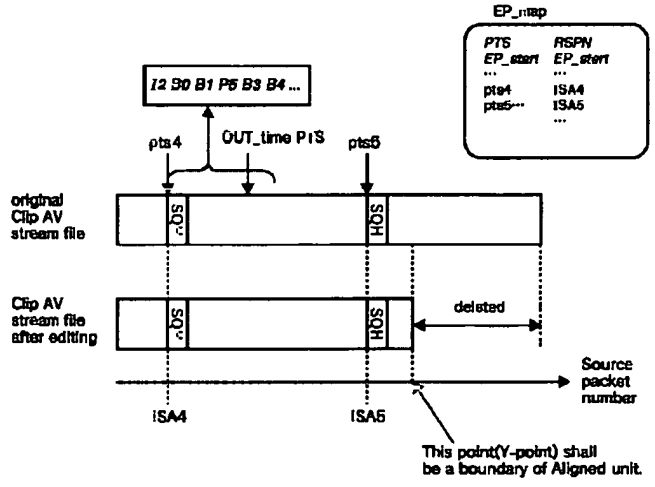
図99と比較の為に、所定の一定ビットレートのトランスポートストリームを符号化することによって、AVストリームの時間経過とAVストリームのデータバイト量との関係が、比例することを保証する符号化モードを説明するフローチャート。
テープメディア（D…VHS等使用されている方法）

【図112】



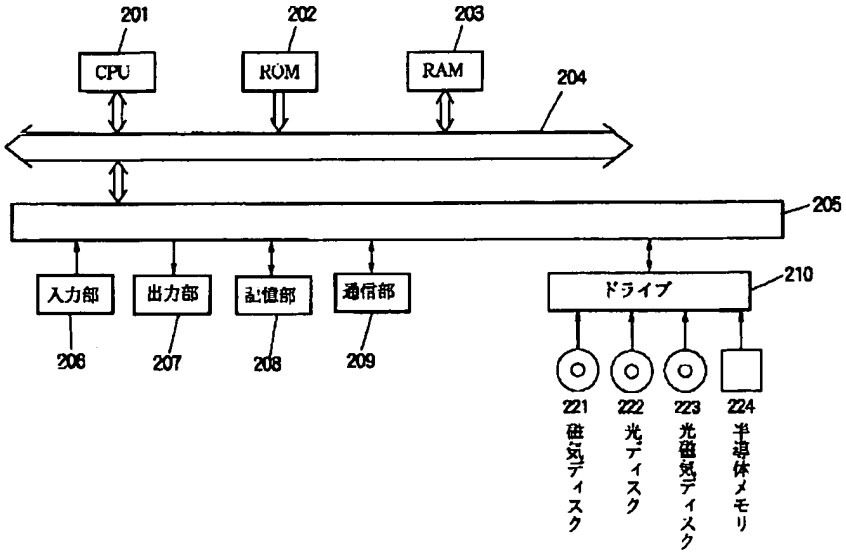
ミニマイズの時に IN_time の前の不要なストリームデ -タを消去する例

【図113】



ミニマイズの時に OUT_time の後ろの不要なストリームデータを消去する例

【図115】



フロントページの続き

Fターム(参考) 5C053 FA06 FA15 FA17 FA20 FA24
GB05 GB38 HA24 HA25 JA01
JA22 JA30 KA01 KA07 LA07
5C059 KK22 KK35 KK36 MA00 PP05
PP06 PP07 RC07 SS13 TA60
TB03 TC03 TC10 TC16 TC18
TC38 TD03 TD06 TD11 UA02
UA05 UA38
5D044 AB05 AB07 BC04 CC06 DE49
EF03 EF05 GK08
5J064 AA00 BB08 BB10 BC02 BC21
BC25 BD03